INTERNATIONAL
HELLENIC
UNIVERSITY

Blockchain anonymity for tracking purposes, within EU, in the era of pandem-ics

Stella Dimitsaki

SID: 3308190005

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Data Science

JANUARY 2021

THESSALONIKI – GREECE

Blockchain anonymity for tracking purposes, within EU, in the era of pandemics

Stella Dimitsaki

SID: 3308190005

Supervisor: Dr. Stavrinides Stavros

SCHOOL OF SCIENCE & TECHNOLOGY

A thesis submitted for the degree of

Master of Science (MSc) in Data Science

JANUARY 2021

THESSALONIKI – GREECE

Abstract

This dissertation was written as a part of the MSc in Data Science at the International Hellenic University. Pandemics are a critical problem in our society these days. Several technology solutions are used to confront this problem. One of them is contact tracing apps. These apps usually face data privacy issues. The latest European Union legislation for users' data protection comes into conflict with the existing contact tracing apps. The purpose of this study is the development of a contact tracing application that follows the General Data Protection Regulation (GDPR). This application is designed with Block-chain technology and it is deployed with python Django framework. Finally, this thesis evaluates the success rate of the app according to GDPR and further discusses challenges that are rose during the development process.

I would like to thank my supervisor, Dr Stavrinides Stavros, who provided invaluable guidance and insightful feedback throughout my research.

Stella Dimitsaki

3/1/2021

Contents

1 Introduction

In the last 100 years, we have faced many pandemic outbreaks, which resulted in world-wide economic instability and social repercussions. Nowadays, the world is struggling to contain the largest epidemic in decades. Due to the ever-increasing connectivity across the world, infectious diseases pose significant threats to international travel. Large gatherings and larger populations have increased the chances of transmission from person to person, country to country, and continent to continent. As a result of the highly transmissible environment, constant disease surveillance is paramount. The first and maybe most vital disease surveillance stage is the accurate and truthful reporting of all suspected and confirmed cases.

An effective way to accurately report the spread of a pandemic is by contact tracing apps. Cellphones contain all kinds of information about where users have been, what they have been doing, and whom they have come in contact with. The use of mobile phones to track disease has gone even further in recent years as a result of 86 percent of the world's population being connected to mobile coverage. There is a plethora of contact tracing apps, which are developed with different technologies and they also use various ways to alarm the users for the pandemic spread.

The University of Oxford estimates that approximately 60 percent of the population would need to use a tracing app to stop an epidemic.[61] Diversely, a recent poll by Washington Post and University of Maryland found among Americans with smartphones that only 40 percent said they were willing to use a contact tracing app citing distrust of big tech companies. One possible way to increase app use is to find a balance between privacy and technological surveillance. [52]

Contact tracing apps use a significant amount of users' data. For this reason, devel-

opers of these apps have to be very careful with the technologies used for this. More specifically, in accordance with EU's involvement with cyberspace [34], recent legislation for protecting personal data in Europe (GDPR), which comes into force on May 25th of 2018, defines personal data as any information relating to an already identified individual or data that can identify an individual either directly or indirectly. [77] As a result, the information that a contact tracing app collects for a user comes against the General Data Protection Regulation (GDPR) because these data can be used to identify any person that use this app.

The problem mentioned above could be bypassed with the use of blockchain technology. Blockchains are incredibly popular nowadays. A blockchain is a distributed ledger that is that is immutable and open to anyone. Once some data has been recorded inside a blockchain, it becomes expensive to change due to the high computational power required to do so. [59] Blockchain features include a distributed structure, auditability, immutability, security, and anonymity. [20] However, there are considerable implications between blockchain technology and GDPR legislation.

An Ethereum smart contract technology could provide a decentralized blockchain solution for a secure contact tracing app database. Ethereum is an open software platform based on blockchain technology that enables developers to build and deploy decentralized applications known as Dapps. Smart contracts develop Ethereum dapps. A smart contract is a computer program stored inside a blockchain as a cryptographic "box". More precisely, developers could deploy their dapps on sequences of logical expressions by using a Turing-complete programming language, known as Solidity.[10]

1.1 Problem description

This dissertation aims to declare the need for contact tracing in a pandemic crisis that follows the EU legislation for data privacy. Blockchain technology is recommended as an approach to the development of a contact tracing app.

Firstly, it is essential to note the usefulness of contact tracing apps during a pandemic. For this purpose is presented a comprehensive literature review on contact tracing apps that were already published in several countries. Moreover, it is a prerequisite to study the structure of these applications by describing in detail their architecture and their technology.

Next, as the blockchain is complicated as it is recently introduced in the technology field, it is crucial to explain the critical elements. After presenting blockchain processes and architecture, it is noteworthy to list the potential applies to this technology.

As soon as it concerns the EU legislation, the GDPR raises concerns about how blockchain technology could follow this regulation. Consequently, there are substantial challenges that need to bypass and be combined with possible solutions.

Finally, the architecture of the application will be designed based on the requirements of the experiment and analyzed the tools used for it. Additionally, it is critical to specify the effectiveness of the application and its weaknesses.

1.2 Dissertation guide

The dissertation is organized into seven sections. These are the introduction to the study, the literature review that reports the two basic parts of the final solution, the technical methodology that is followed for the experiment, and the conclusion and the recommendations.

Section one includes the introduction to the necessity of contact tracing in the era of pandemics. Furthermore, it mentions how mandatory the compliance of contact tracing application to the new GDPR legislation and proposes the smart contract blockchain technology as a possible solution.

Section two analyzes the contact tracing use by detailing a description of the system's architecture and technology. Moreover, it lists the existing contact tracing apps worldwide and is discusses their strengths and their weakness.

Section three specifies the basic structure of a blockchain. It notes blockchain's architecture, functionalities, and applications. It also enumerates the different types of blockchain. In addition, it presents the challenges of blockchain and GDPR, and it proposes optimal solutions. This section mentioned the existing approaches for contact tracing blockchain applications.

Section four involves an extended description of Ethereum's smart contract technology, and it explores the technology tools that will be used in the experiment.

Section five provides a completed view of the application's user interface for the hospital's view and the supplemental application for the tracing view. It also comments on the programming code and its functionalities.

Section six discusses the results of the experiment. Furthermore, it mentions the challenges faced during the deployment of the application.

Finally, section 7 concludes how the solution proposed to the problem is the most effective. Moreover, the benefits and limitations of the app are listed. Lastly, there are several suggestions for future work.

2 Tracking needs in Contagious pandemics

The development of mechanisms for efficiently controlling the spread of infectious diseases is a significant part of mathematical epidemiology. Contact tracing is an important tool for handling the extension of a pandemic. [54]

Contact tracing is a system that has been used for decades to stop infectious diseases. A contact is anyone who has had direct physical contact or was within one meter for at least 15 minutes with an infected person even if that person does not have symptoms. The tracing applies to anyone who has had contact from two days before a person gets sick. Once an individual has been confirmed as a contact, the authority responsible for the spread of the pandemic will be asked the contact to go into quarantine willingly. The conditions mentioned above imply the separation from others for a particular time until it is clear that the contact has not any sign of the illness.

Recently, the awareness about the impact of density on emerging highly contagious infectious diseases has been increased. More specifically, dense areas are more compacted, making them possible hotspots for the rapid pandemics transmission. [25] Contact tracing could be used to provide valuable information to members of those crowded communities, and it would prepare them to control the spread of a pandemic effectively. [16]

To sum up, contact tracing provides a plethora of solutions to the spread of an epidemic. Firstly, identifying and limiting contacts of people infected with the virus stops the transmission, and the community stays safe. Sequentially, this tool helps epidemiologists have a clearer view of how an epidemic is spreading. Lastly, scientists could also do more accurate research with this valuable information to develop a more accurate solution for

facing a pandemic.

2.1 Contact tracing apps

Contact tracing has been essential for fighting the spread of novel diseases. The architecture of these apps and their functionalities are changing according to the country and its legal system. Some countries have more rigorous regulations on privacy than others. [12] [60]

2.1.1 System Architecture

There are three distinct architectures of contact tracing apps based on the privacy systems and how personal data is stored on the server. Firstly, there is a centralized architecture where data is concentrated in a server. Secondly, there is a decentralized architecture where the data saved in the device of every individual. Finally, there is a hybrid architecture, which is the combination of the architectures mentioned above. [3]

The examples illustrated below are a contact tracing app for the COVID19 virus that led to the development of many contact tracing apps the last year.

Centralized

Figure 1 presents a centralized architecture model. It is a graphic representation of entities and interactions in the Bluetrace protocol, which follows a centralized model. Bluetrace protocol is used for contact tracing apps that are deployed with Bluetooth technology to protect users' data. [8] Firstly, the app requires the user to pre-register to the central server. Sequentially, the server will provide a temporary ID(TempID) for every device that will be encrypted with a secret key, and it will be sent to the device. These ids will be exchanged between the devices that their users come in close contact with. Finally, when a user willingly uploads his status after testing positive for a virus, the central server will

alert the users with the TempID stored as contacts to the infected user.[3]

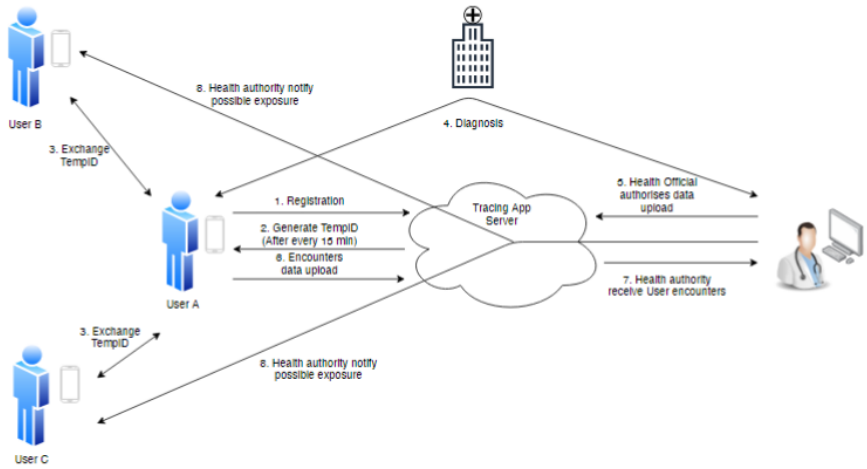

Figure 1: Tracing apps centralised architecture [3]

Figure 2 demonstrates the process that a user needs to follow to register in a centralized architecture. After downloading the app and sign up by providing the necessary personal information, the user needs to verify his mobile number by One Time Password (OTP) procedure via SMS that the server sends. Subsequently, the server generates the TempID for the user's device.[3]

Figure 3 shows the exchange TempIDs procedure between users' devices. The procedure of exchange is implemented with the use of "Encounter Message" and Bluetooth technology. The encounter message includes TempID, Phone Model, and Transmit Power(Tx-Power). Also, the user's device stores the Received Signal Strength Indicator (RSSI) and the timestamp of the message delivery. The Bluetrace protocol uses a blacklist method to avoid duplicate records in the user's device. This method blacklists encounter messages that are already received from a contact's device for a short time. [3]

Figure 4 details the execution path that the application follows after a user tests positive for the virus. The clinician reports the user as infected to the server, and the server performs

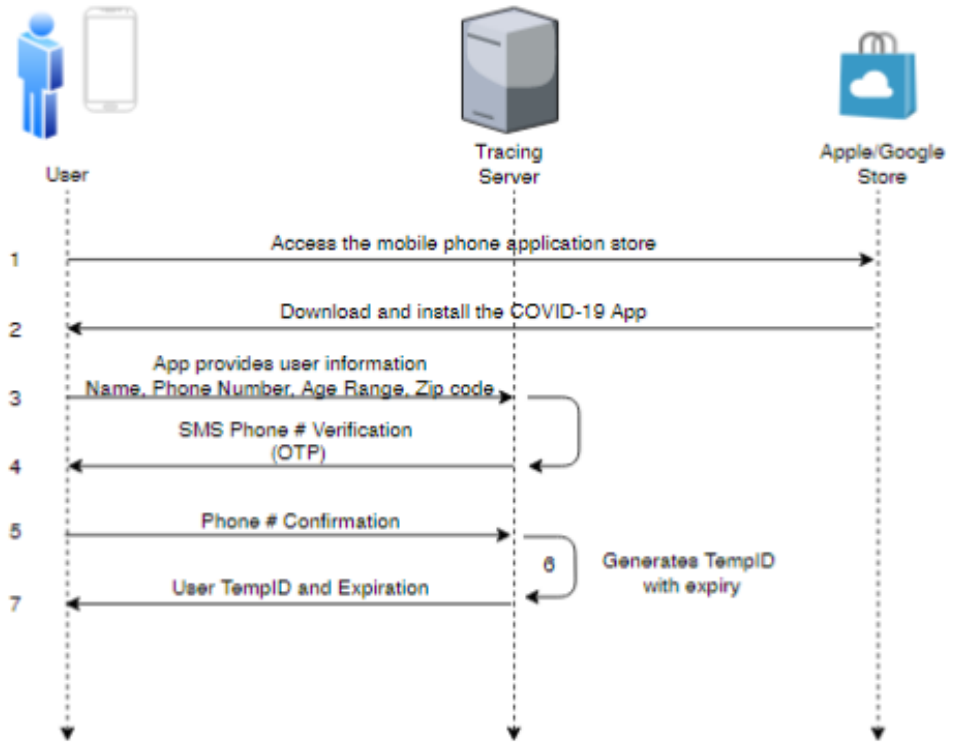

Figure 2: Centralised tracing app registration process [3]

an OTP procedure for verification that permits data to be stored. Sequentially, the server located the encounter messages and TempIDs stored, it unencrypted them with the private key and informed the clinician about the infected user's contacts. Conclusively, the health official notifies the suspected contacts. [3]

To sum up, in centralized architecture, the main procedures are executed from the central server. The authorization that is given to the server by this architecture may cause privacy issues.

Decentralized

Controversially, to the centralized architecture, the decentralized architecture introduced a procedure where there is no central server with authority to implement the main

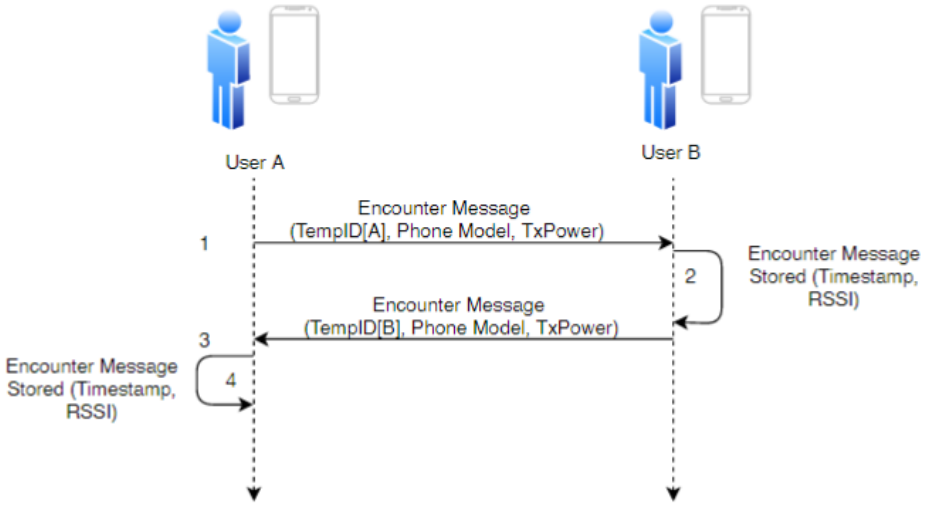

Figure 3: Centralised tracing app contact exchange operation [3]

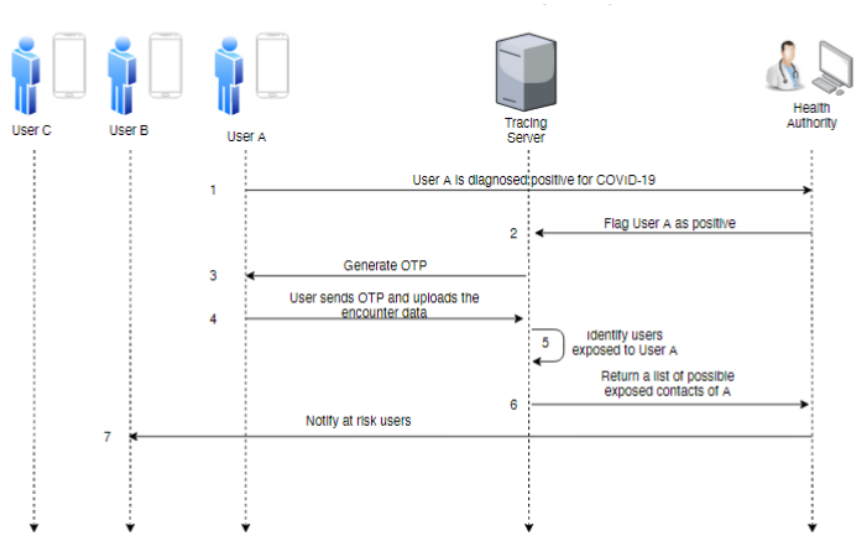

Figure 4: Centralised tracing app notification [3]

functionalities. More specifically, most of the decentralized contact tracing app processes take place on the user's device. This implementation secures the user's data by generating anonymous identifiers at the user devices and controlling the exchanged notifications on

user devices instead of the centralized server. Figure 5 illustrated the Private Automated Contact Tracing protocol (PACT) graphic representation, which is used to describe the decentralized architecture structure. PACT is developed to improve the data privacy in contact tracing apps during a pandemic by designing exposure detection functions implemented in personal digital devices. [69]

At the decentralized approach, the pre-registration of the user is not required. By using a pseudorandom function, devices generate seeds. Seeds combined with the current time create privacy-preserving pseudonyms or 'chirps' with a very short lifetime of about 1 min. These chirps are transacted between users' devices whenever they have contact.

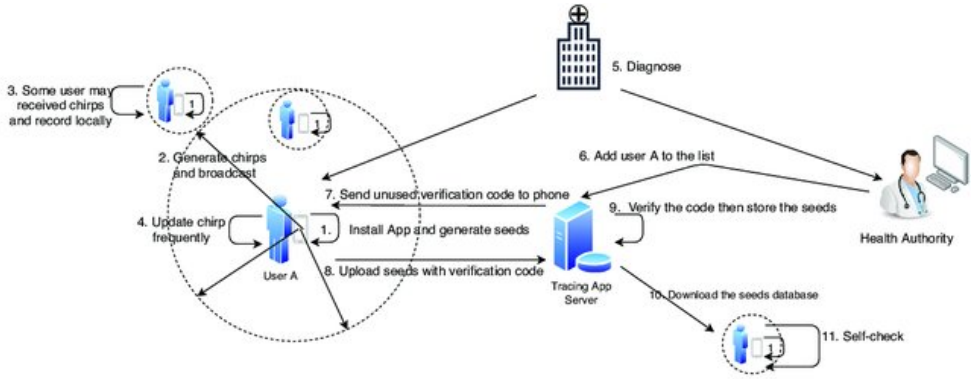

Figure 5: Tracing apps decentralised architecture. [3]

Contact tracing apps designed as decentralized are not required to register users after the installation process. It can be seen in figure 6 that after the app is installed on the device, a random seed generation is deployed.

As mentioned above, the seed produced and the current time are consequently used in a pseudorandom function to generate the chirp. The chirps are not connected to an individual or their mobile device, so they are anonymous. In figure 7, it is depicted that chirps are transmitted every few seconds via Bluetooth. The chirp received from the contact phone is stored in the app accompanied by the timestamp that the chirp is exchanged and the

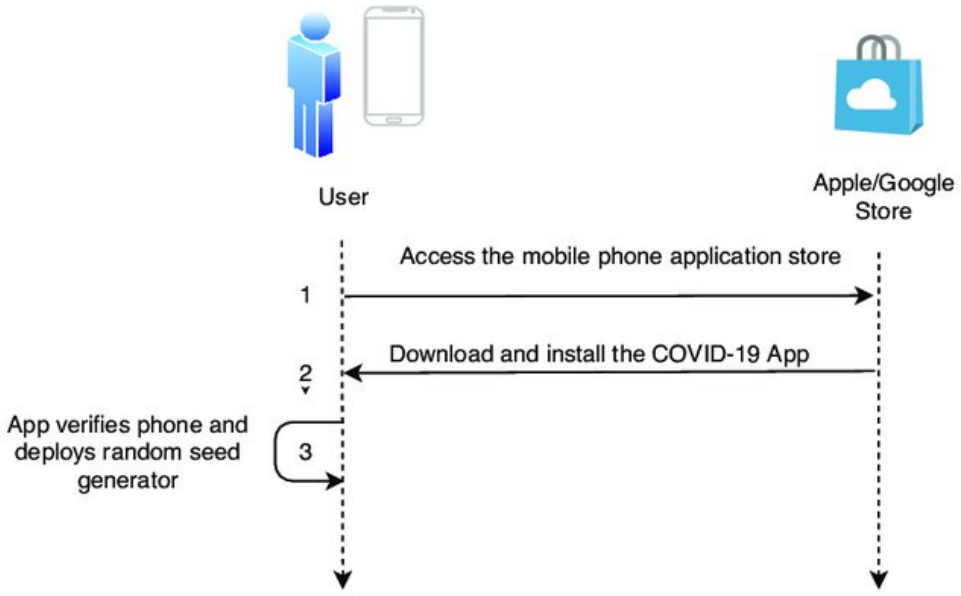

Figure 6: Decentralised tracing app installation process. [3]

maximum Received Signal Strength Indicator (RSSI) value. For avoiding the exchange of chirps more than one time with the same contact, chirps that are received within 1 min, are ignored.

After a user is tested positive for a virus, the procedure is that the health official will give a unique 'permission number' to the infected user, as shown in figure 8. This permission will authorize the upload of used seeds recorded to the mobile device along with the creation and expiration times of the seeds. In contrast to the centralized approach, the server only receives the seeds linked to a single identified user in the decentralized approach.

In conclusion, decentralized architecture executes the core tracing procedures locally to devices of users. The central server is used only once per day to download infected individuals' seeds to mobile devices. After downloading, the contact tracing app reconstructs all the corresponding chirps.[3]

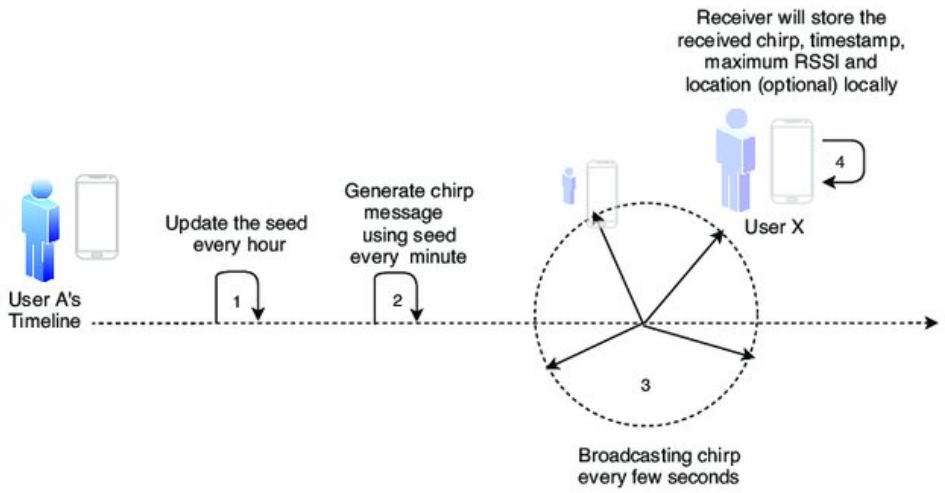

Figure 7: Decentralised tracing app encounter exchange. [3]

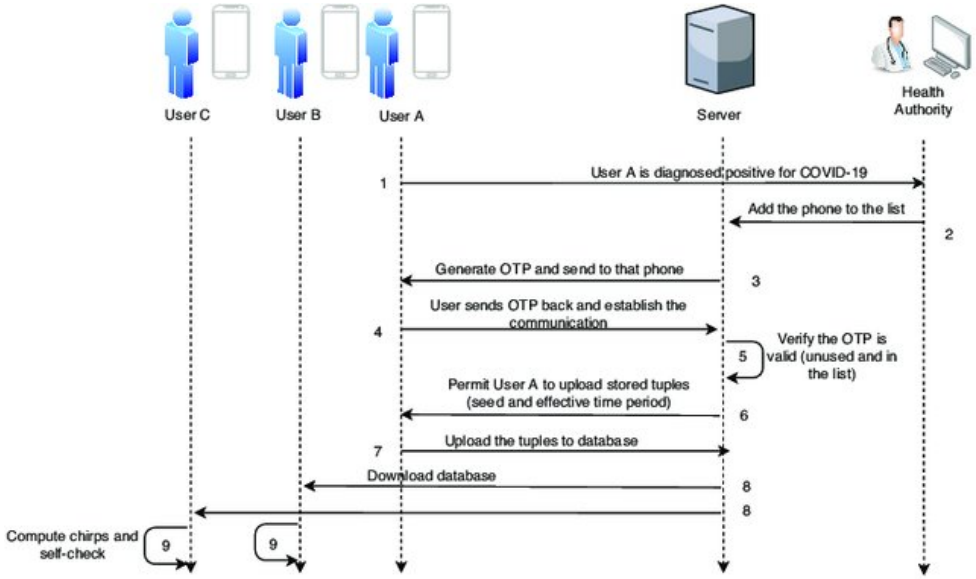

Figure 8: Decentralised tracing app tracing process. [3]

Hybrid

There is also the hybrid architecture, which is a combination of centralized and decentralized. In the hybrid approach, the core functionalities are distributed in the server

and the devices. While the risk analysis and notifications are remaining centralized, the generation of TempID is performed decentralized.

Figure 9 illustrates the entities and the interactions of the Desire protocol. [11] User registration is a requirement at this protocol to generate an Ephemeral ID (EphID) that will be exchanged with the contact devices. After the transaction of EphID, the mobile device produced and stored two unlinkable Private Encounter Tokens (PETs) to represent the contact. By the time a user is reported by a health official as infected by the virus, the PETs stored in the device are sent to the server. Any user can upload the second PET to the server, which then performs risk analysis and notification.

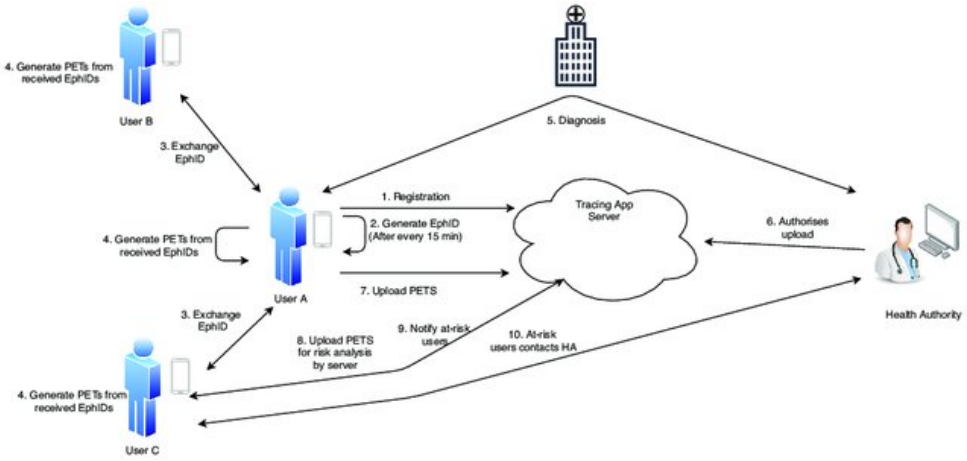

Figure 9: Decentralised tracing app tracing process. [3]

Figure 10 presents the two-step authentication process through OTP that the user has to follow after downloading the app to register. Thus, the server verified the app through an authorization token and deleted the user's phone number. Subsequently, the server generates a unique ID and an encryption key to send to the app. Finally, the encryption key is deleted from the server.

The mobile device forms an Ephemeral ID (EphID) by using the Diffie Hellman key exchange mechanism. [18] The EphID is active for 15 min, and it is synchronized with

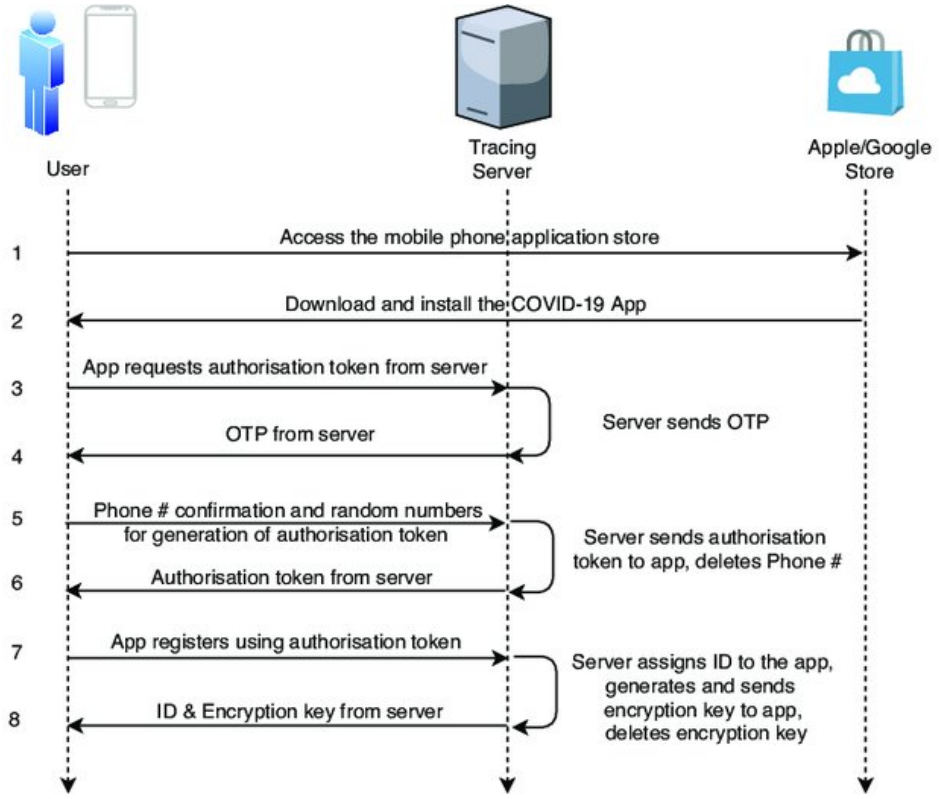

Figure 10: Decentralised tracing app tracing process. [3]

the Bluetooth MAC address rotation interval, as shown in Figure 11. Once the EmpID is exchanged with another device, the app produces two PETs. These are stored in the two tables that the app retains, one in the query table and the other in the upload table with the contact's time and duration.

Figure 12 details the process after a patient is tested positive for the virus. The user uploads ID, encryption key, and PETs in the upload table along with time and duration values willingly. Subsequently, the server maintains these PETs combined with data. The use of the encryption key updates the values in the user's record.

Whenever any user chooses to check their contacts for infected cases, the device uploads the PETs stored in the query table to the server. Next, the server iterates in the upload

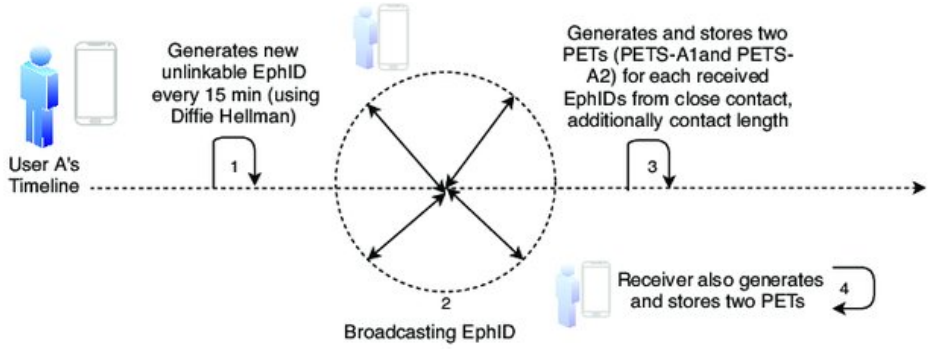

Figure 11: Decentralised tracing app tracing process. [3]

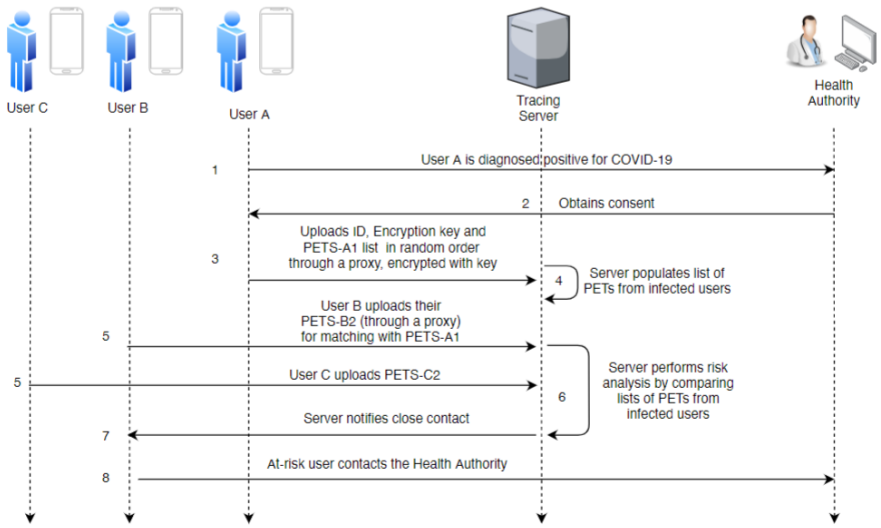

Figure 12: Hybrid tracing app notification process. [3]

table to find the PETs match from the query table. Thus, it performs a risk analysis based on the time and duration values and notifies the user. It is essential to mention that the server cannot identify any user by the PETs values.

In conclusion, Figure 13 presents the architecture distribution of the published contact tracing apps worldwide. As we can see, most of the apps follow the decentralized architecture, and less of them the Hybrid architecture. A significant percentage of the apps follow

the centralized architecture, too.

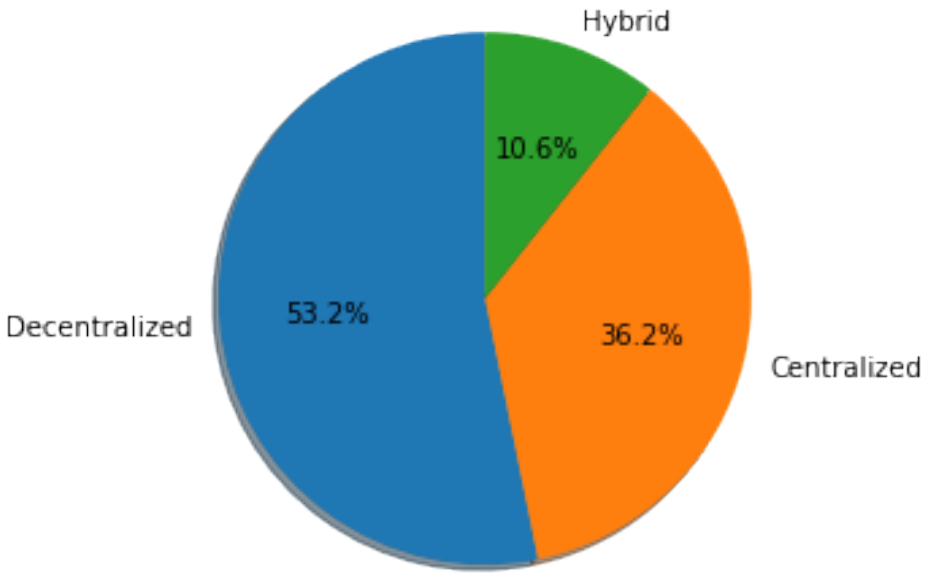

Figure 13: Architecture distribution of contact tracing apps.

2.1.2 System Technology

Most of the contact tracing apps are developed with Bluetooth or Geo-location technology. Figure 14 presents the distribution of these technologies in published applications.

Bluetooth

In contact tracing apps that have been developed with Bluetooth technology, the installation of the app is necessary. After the user came in contact with a virus carrier, who also had installed the app, the application will store the information, and it will notify the user about the risk. [73]

This technology has the following strengths:

- Bluetooth is more efficient in short proximity.

- Every device is anonymized by using a random key.
- The application works without any cellular data.

On the other hand, Bluetooth has significant limitations.

- The contacts of the user must have already installed the application.
- The user's device is necessary to be a smartphone with a Bluetooth sensor.
- Bluetooth technology has privacy issues as it can be used as a portal to mobile device data.
- This methodology can produce false-positive outcomes.

Location

No application is needed to be installed on the user's device at the Geo-location type of application. All the necessary information is collected via the SIM (Subscriber Identity Module). At this approach, active government support is required. [73]

There are a plethora of advantages to the geo-location approach. The most significant are the following:

- The user's device does not have to be a smartphone. In addition, it is not essential to have a Bluetooth sensor.
- SIM technology exists in every mobile phone. Thus, this type of contact tracing refers to all mobile phone holders.
- The information about the location is always correct, and it is highly improbable to fail.
- The number of false negatives is inconsiderable.

There are also significant limitations to this contact tracing model.

- The user has to avoid sharing the mobile device with other users because the location record will keep inaccurate data. As a result, the user may be notified about an infectious contact that never really happened.
- The geo-location abides the same on every floor of a building.

Some applications are developed with both technologies. As shown in figure 14, they cover a substantial piece in the market of contact tracing apps.

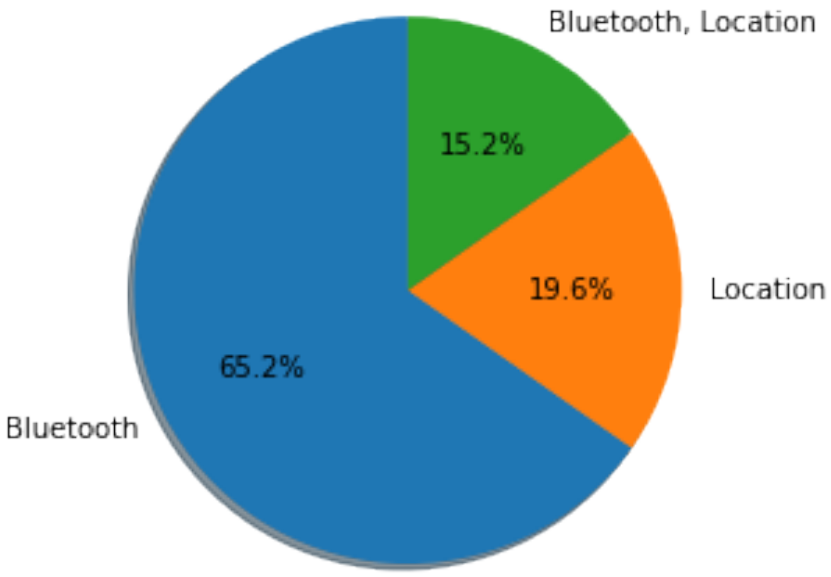

Figure 14: Technology distribution of contact tracing apps.

2.1.3 Applications

After the recent COVID19 pandemic, most countries are creating their contact tracing apps to manage the spread of the virus. Most of the apps are centralized, and they combined

both Bluetooth and location tracking technology. As figure 15 indicates, more than 50% of the apps are used voluntarily by citizens.

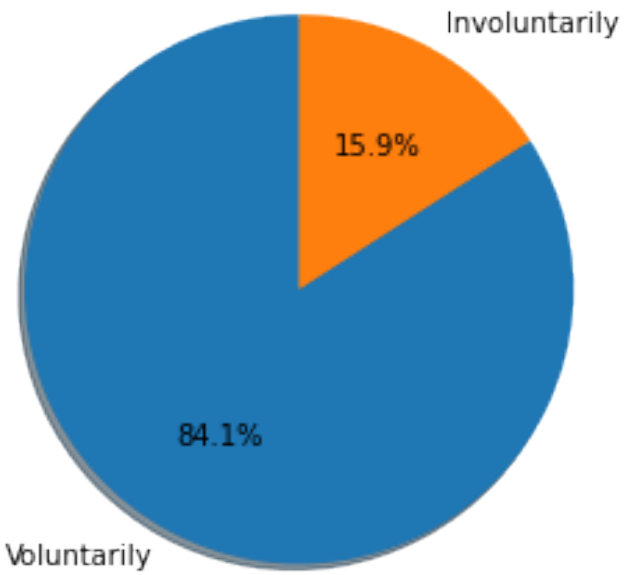

Figure 15: Voluntarily distribution of contact tracing apps installing.

Below are some examples of countries which are currently utilizing contact tracing app technology:

China

The Chinese government and Alipay developed Chinese applications. It uses location technology for tracking. It is mandatory for citizens of China to download the contact tracing app. Also, it is not specified whether the data collected from this application is used only for contact tracing purposes and destroyed after the required time. The information that exists for the technology of China's app is limited. Finally, the data is not anatomized [46]

India

India established an application named Aarogya Setu. The app is designed with both Bluetooth and location technology. Same as China, India made use of the app involuntary for the citizens. The data is concentrated in a centralized server and is used only to protect public health, but it is not mentioned clearly if the data is deleted after a certain time. There is no anonymization of the users. [24]

Indonesia

PeduliLindungi is the contact tracing app of Indonesia. The Indonesian app uses users' location data that is provided by telecommunications companies. Sequentially, the application is centralized, and the data is gathered via Bluetooth and location technology. There are significant privacy issues because it is not specified if the data is erased after its use or analyzed for other purposes. It also voluntarily. [14]

Japan

COCOA is the app launched by the Ministry of Health, Labour, and Welfare of Japan. Because of the plethora of issues, it has been suspended a couple of times. It is produced with Bluetooth technology, and it is decentralized. There are no significant data privacy issues because the data is removed after a determined amount of time, and it is only used for the app. [37]

Mexico

Mexican government launched the CovidRadar, an application that is centralized and is based on Bluetooth technology. The use of data apart from the application is unspecified, and there is not anonymization. Conclusively, privacy and data policies are questionable.[81]

Philippines

Multisys Technologies Corporation developed the StaySafe application for the government Philippines' government, decentralized and used Bluetooth technology. The installation of the app is voluntary, and the users are not anonymized. The app has serious privacy

concerns because it does not clearly mention the management of the data gathered on the server. [15]

Vietnam

BlueZone, the contact tracing app of Vietnam, was designed with Bluetooth technology. Although it has decentralized architecture, it requires access to contacts and other media on mobile devices. Individuals that download the app remain anonymous to the application's data. Information that is concentrated on users' behavior is not erased after a short time. [75]

Germany

Deutsche Telekom and SAP launched the Corona-Warn-App for the German government. The application was produced with Bluetooth technology. Despite the fact that the first option was a centralized application, the government concludes with a decentralized approach for privacy reasons. Data is anonymized and, after its use, deleted. [68]

Turkey

Turkish Ministry of Health created the contact tracing application Hayat Eve Sığar. The patients who detected positive for the virus must install the app on their mobile phones and share it with the police. Subsequently, data is not anonymized. Even though the gathered data erased after its use, it is not determined if it is consumed only for public health purposes. [32]

Iran

The Sharif University of Technology in Iran developed a location technology application, the AC19. The app is not mandatory for citizens. It is important to mention that the app is forbidden by Google Play because it collected data that is not essential for contact tracing. In addition, the application was transmitting real-time location that was not anonymized to the state. [48]

France

Although France negotiated with Apple and Google to develop the contact tracing application, it completed with an application by the National Institute for Research in Digital Science and Technology, Inria. TousAntiCovid is created with Bluetooth technology, and it follows the decentralized architecture. There are no privacy issues because the application is GDPR regulated. [53]

UK

NHS launched a COVID-19, Bluetooth, and decentralized contact tracing app in the UK. The application secures users' data by anonymization and deletion after its use. Additionally, the app serves only public health purposes. [80]

Italy

Bending Spoons is the Italian company that is constructed the contact tracing app Immuni. The installation is voluntary for the residents. The application has no privacy issues because it is used strictly for public health objectives, and data is anonymized. [43]

Poland

ProteGO is modeled after efforts in Singapore by the Ministry of Digital Affairs of Poland. It is a Bluetooth application that follows a decentralized architecture. Users' data are secure by anonymization. [51]

Saudi Arabia

Ministry of Health and the Saudi Data and Artificial Intelligence Authority launched two contact tracing applications. The first one, Tabaud, is modeled to allow citizens to request "movement permits" for mobility around the city. It is decentralized, and it uses Bluetooth. The second one Tawakkalna is tracking the transmission of the virus. There is limited information about the privacy issues of the app. In both applications, downloading is not mandatory for the residents. [26]

Malaysia

MyTrace is a Bluetooth Malaysian app modeled by the Ministry of Science, Technol-

ogy, and Innovation, and it is available voluntarily only on Android. The government has announced to publish the open-sourcecode. The application is decentralized, and it does not store anonymized data. Thus, the security of users' data is problematic. [56]

Ghana

The Ministry of Communications, in partnership with iQuent Technologies Ascend Digital Solution, has developed the GH COVID-19 Tracker App. It collects location data of individuals that are not anonymized or destroyed when it is not used by the app anymore. As a consequence, there are privacy issues in the app. The installation of the app is not mandatory. The architecture of the app is not clearly mentioned. [71]

Australia

COVIDSafe is a Bluetooth, decentralized contact tracing application that is created by the Australian government. Although collected data are used only for public health reasons, they are anonymized and deleted after a determined amount of time. Australian experts have criticized the government for an absence of transparency and unresponsive to data privacy issues. [78]

Tunisia

Tunisia's Observatory of Emerging Diseases constructed E7mi. The privacy policy of data is not noticed precisely. It is known only that data is used for public health purposes, and they are erased ever after. Although the app download is voluntary, the government announced that it would remain voluntary so long as download rates remained high. [79]

Czech

The Czech Ministry of Health launches the eRouska app as part of the government's "smart quarantine" plan. It is a decentralized, Bluetooth application that follows all the privacy rules that are compliant with GDPR. [27]

Hungary

The Hungary Ministry of Innovation and Technology and the NextSense company de-

veloped VirusRadar. The application is optional. It is designed as centralized, and it collects data via Bluetooth. The center server saves individuals' contacts that last more than 2 minutes, and it drops them after 20 minutes. [29]

UAE

TraceCovid is a decentralized, Bluetooth application that is constructed by the Department of Health. Citizens of the UAE must install the app on their mobile devices otherwise will be fined. The use of stored data is not specified clearly. Also, there is no anonymity. [67]

Israel

HaMagen is the contact tracing app published by the Health Ministry of Israel. Because the application is using location technology based on GPS, officials supported is not sufficiently accurate. The architecture is centralized. The central server is securing the data by anonymization, and it is erased after its use. [65]

Austria

The Austrian app Stopp Corona launched by Red Cross was one of the first major European nations to align with the Google/Apple API. The application is decentralized, and it is working with Bluetooth technology. Data are processed only for public health reasons, and they are anonymized. Moreover, the collected information is deleted after its use. [4]

Switzerland

The first option of the Swiss National Covid-19 Science Task Force for the development technology of SwissCovid was the Decentralized Privacy-Preserving Proximity Tracing (DP-3T) instead of the Google/Apple API. Eventually, they used both technologies. Furthermore, data is gathering by Bluetooth. There are no privacy issues. [17]

Bulgaria

Scalefocus company created the ViruSafe, and it passed it to the Bulgarian Ministry.

By using location technology, the application generates a map with possibly infected individuals. Even though the application requests users' data, users' identity is secure by anonymity. The time limit that data is deposited in the centralized server is undefined. [35]

Thailand

Digital Government Agency of Thailand launched a decentralized app MorChana. The application combined the proximity contact tracing app with a QR code check-in system, called Thai Chana. The privacy of the data that is gathered by Bluetooth and location is not specified in detail. [28]

Singapore

Government Technology Agency launched the first major Bluetooth contact tracing app. Furthermore, the BlueTrace protocol is introduced by TraceTogether. The installation of the app is mandatory for Singapore's citizens. The architecture is centralized. [41]

Finland

The Finnish government approved the development of the Koronavilkku contact tracing app by the Communicable Diseases Act. The application is voluntary and free. The data is gathered via Bluetooth technology. [50]

Norway

Smittestopp is a centralized application that was published by the Norwegian Institute of Public Health. Although it was notified that data is received only for public health purposes, Norway withdrew the app and destroyed all user data because of security concerns and criticisms. [57]

Ireland

In contrast to the neighboring UK, Ireland opted to use the Google/Apple API that is based on Bluetooth technology. Covid Tracker is decentralized, and Ireland's Health Services launch it. The application secures users' data with anonymization. Data also

erased after a determined amount of time. [44]

New Zealand

New Zealand's Ministry of Health created a centralized contact tracing app with Bluetooth technology. NZ COVID Tracer is premised on a check-in system using QR codes in public areas. The privacy policies for user's data are not specified accurately. [30]

Kuwait

Shlonik is a centralized contact tracing app that is developed by Kuwait's Ministry of Health. The application concentrated data based on individuals' real-time location. An Amnesty International report highlighted this app as one of the most invasive in the world. [70]

Qatar

Ehteraz is a contact tracing app that is designed with both Bluetooth and location technology. The Ministry of Public health publishes it. The application requires access to photos and is not voluntary. In addition, it is detected a security hole in the database scheme. [66]

North Macedonia

North Macedonia uses the StopKorona contact tracing app that is developed NectSense company like Hungary's app. It is decentralized, and it is collecting data via Bluetooth. [36]

Northern Ireland

Northern Ireland has closed a deal with the same developer as Ireland's app. StopCOVID NI is a decentralized, Bluetooth application. There is not sufficient information about data security. [9]

Bahrain

The Information and eGovernment Authority of Bahrain designed a centralized application that is gathering data without anonymization. BeAware is voluntary. As a result,

the number of downloads is not more than 25% in the country. [7]

Estonia

HOIA, the Estonian contact tracing app is developed with DP-3T and Bluetooth technology borrowed by Google/Apple API. The app's development resulted from Bytelogics, Cybernetica, Fujitsu Estonia, Guardtime, Icefire, Iglu, Mobi Lab, Mooncascade, and Velvet corporation. For data collection is selected the decentralized approach. [38]

Cyprus

CovTracer is the Cypriot contact tracing app launched by the Research Centre of Excellence on Information and Communication Technologies (RISE). It is collecting data via real-time location, and it is storing them anonymized. Although there is no policy about the use of the data. [35]

Fiji

Fiji Government and digitalFiji company published CareFiji. The application is developed according to the Singapore TraceTogether. It is a decentralized Bluetooth app. There is no privacy policy for users' data. [1]

Iceland

Iceland opted not to use Bluetooth because the results may be inaccurate and instead utilizes location technology. Rakning C-19 is a contact tracing apps designed with both decentralized and centralized architecture, a hybrid approach. Individual data are secured and deleted after their use. [31]

Gibraltar

Beat Covid Gibraltar is launched by the HM Government of Gibraltar and the Gibraltar Health Authority. The installation is voluntary, and it is only downloaded from more than 25% of the population. The application is decentralized, and it gathers data via Bluetooth. Data are secured and erased after a determined amount of time. [7]

Algeria

There is no accurate information about Algeria's contact tracing app. It is important to mention that Amnesty International was examined the application for breach of personal data. [62]

Belgium

Coronalert is Belgium's contact tracing app that is designed according to Germany's Corona-Warn app. Interfederal Committee for Testing and Tracing publishes it. The application is decentralized. The data gathered anonymously via Bluetooth, so there are no security issues. [64]

Canada

Both Shopify and Blackberry introduce COVID Alert for the Canadian government. Firstly, it is launched in 8 provinces and territories. Companies developed the application by the use of Google/Apple API and Bluetooth technology. The architecture is decentralized. The collected data are secured. [62]

Denmark

Denmark published Smittestopp, a decentralized and Bluetooth application. Although documentation notifies that data is anonymized and secure, authorities asked for temporary deactivation of its use and deleting the collected data. [2]

To sum up, figure 16 presents the number of downloads for each application. Notably, most of the apps are faced with data privacy issues. As a result, some of them are deprecated after a call from local authorities or Amnesty International.

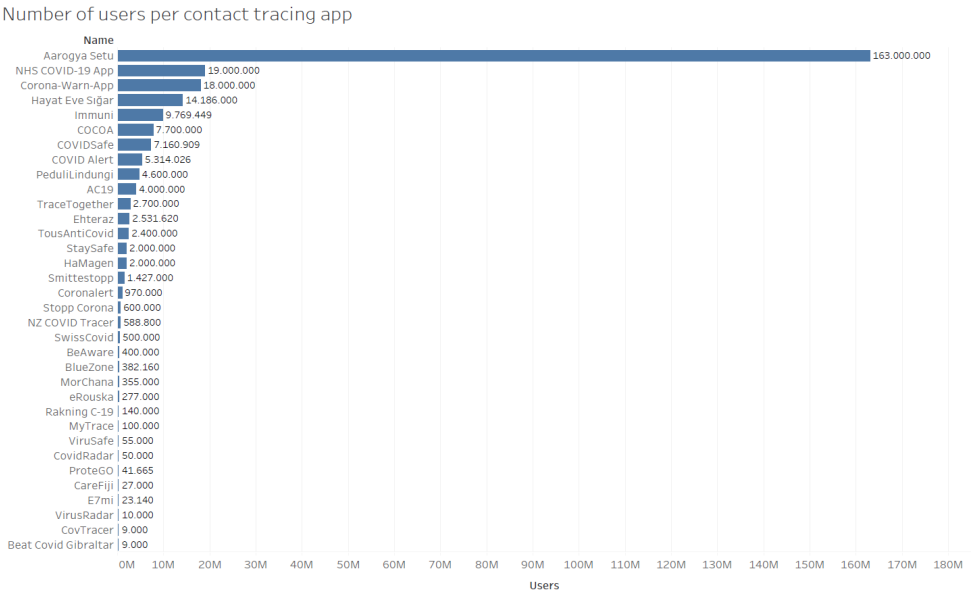

Figure 16: Number of users per contact tracing app

3 Blockchain technology

A blockchain is introduced firstly in 2008 and applied in 2009. The first application of blockchain is the ”bitcoin”, a decentralized peer-to-peer digital currency, with no government to issue it. No banks are needed to manage accounts and verify transactions. [55] Even though Bitcoin cryptocurrency is the most popular blockchain application, blockchain can be used for various applications apart from cryptocurrencies.

3.1 Blockchain architecture

Blockchain could be mentioned as a public ledger, in which all committed transactions are stored in a chain of blocks. This chain continuously expands when new blocks are added to it. Figure 17 presents the blockchain architecture of a sequence of blocks. [55] Each block points directly to the previous one via a reference that is a hash value of the previous block called parent block. The first block in a blockchain is called the genesis block.[84]

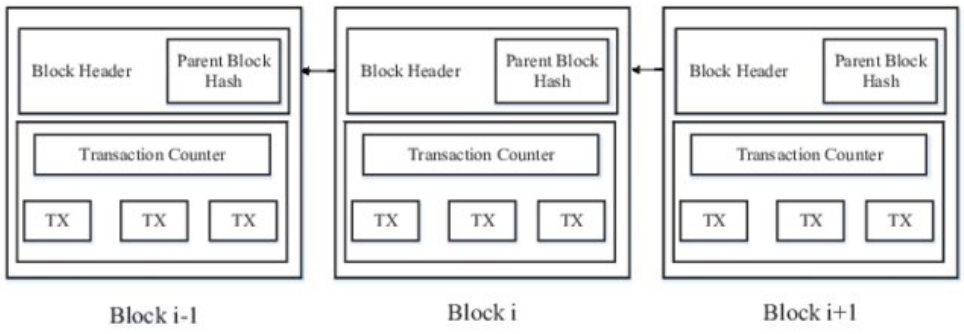

Figure 17: A blockchain architecture with sequence of blocks [83]

3.1.1 Blocks

The data that is stored inside a block depends on the type of blockchain. For example, the Bitcoin blockchain stores the details about a transaction in blocks, such as the sender, receiver, and amount of coins. [63] It can be seen in Figure 18 that the block is separated into two parts the block header and the block body.

The block header has the following values:

- *Block version:* Indicates which set of block validation rules to follow.
- *Parent block hash :* Every block in a blockchain contains the hash of the previous block. This technique effectively creates a chain of blocks, and it is this technique that makes a blockchain.
- *Merkle tree root hash:* Hash identifies a block and all of its contents, and it is unique. A hash is generated when a block is created, and when the block contents change, the hash of the block also changes. If a single block changes, it will make all the following blocks invalid.
- *Timestamp:* Current timestamp as seconds since 1970-01-01T00:00 UTC
- *nBits:* Current hashing target in a compact format [84]
- *Nonce:* A 4-byte field, which is a counter used to make sure each transaction can only be processed once [10]

The block body contains the transaction counter and transactions. Every block can contain a finite number of transactions that depend on the block size. [84]

Last but not least, an asymmetric cryptography mechanism is used to validate the authentication of transactions.[47] Every block contains a digital signature that is used to

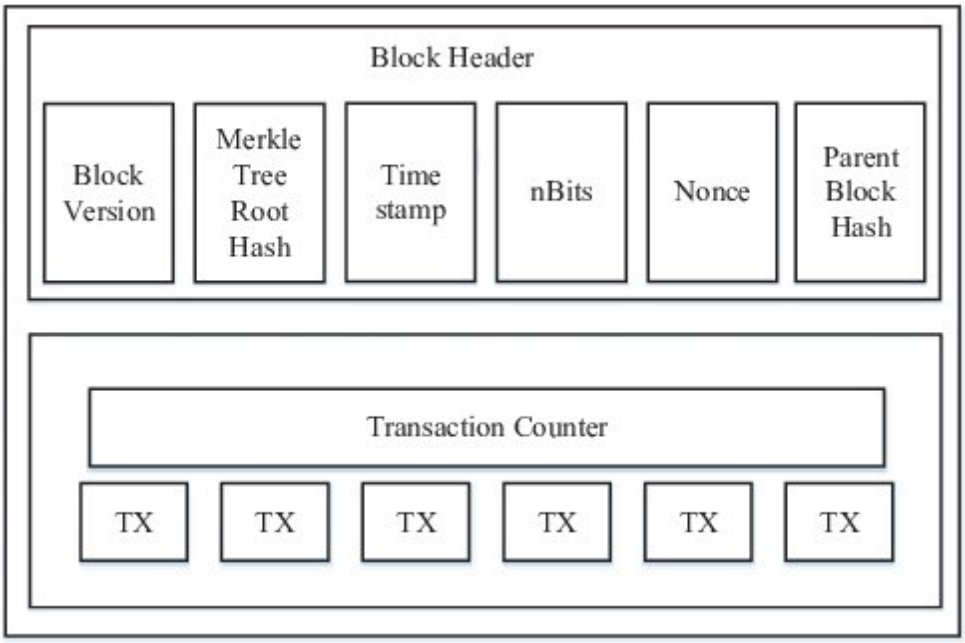

Figure 18: Block structure [83]

sign the transactions. More specifically, a private and public key is provided to each user, and the private key signs the transaction. The Bitcoin digital signature is shown in Figure 19.[21]

3.1.2 Key characteristics of blockchain

The blockchain technologies are composed of key characteristics.

Decentralization

In contrast to centralized transaction systems that each transaction needs to be validated through a central authority, blockchain does not have to rely on a centralized node anymore. The data can be recorded, stored, and updated distributedly. Thus, blockchain can significantly reduce server costs. [84] [47]

Open Source

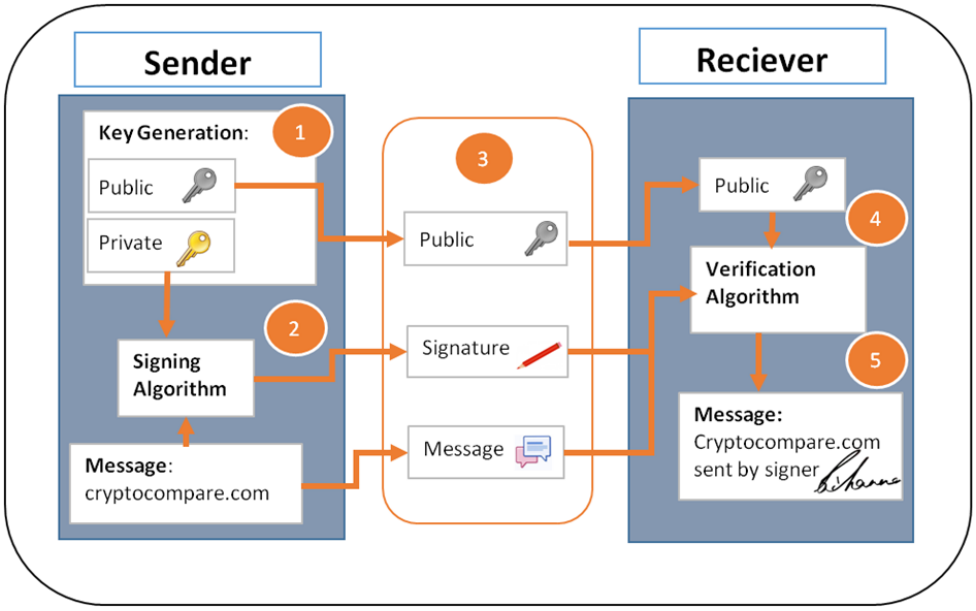

Figure 19: Bitcoin digital signature [21]

Blockchain technologies such as Hyperledger and Ethereum are open to everyone to create any application. [47]

Autonomy

Each node on the blockchain network transfers or updates data safely as a result of consensus. The consensus is based on trust between a single person and the whole system. [47] [84]

Immutability

Records that are stored in a blockchain are unchangeable. This condition only could change if someone can take control of more than 51% of a node simultaneously. [47]

Anonymity

The Blockchain network gives each user the ability to interact with it by generated addresses. As a result, transactions can be anonymous, and a user could generate as many addresses as he wants to avoid the exposure of his identity. [47] [84]

Transparent

The data of every node in a blockchain is transparent. This characteristic leads to the democratization of blockchain data, and it makes the blockchain trusted.

Persistency

Transactions in a blockchain network are distributed in blocks. Also, each block is added to the blockchain after other nodes validate it.

Auditability

Users can trace the previous records of the blockchain network because of every transaction record with a timestamp.

3.1.3 Consensus

The consensus is the name for a group of algorithms that can create a voting process for the network. This voting process helps make decisions about the information on the blockchain. The consensus is known as a solution for the Byzantine Generals Problem. [22] In a distributed system, a consensus is a way to ensure trust between users without any communication between them.

3.1.4 Proof of Work

Bitcoin introduced an innovative way of using Proof-of-Work as a consensus algorithm to validate transactions and broadcast new blocks to the blockchain. Proof of work is costly or time-consuming to produce a piece of data but simple for others to verify the correct solution under precise requirements. There is an upfront cost of resources known as work to generate a block's hash value in proof of work. The rest of the network can easily validate this work to check if it was done correctly. The proof of work algorithm achieves consensus without a central authority. [47]

In addition, each node is involved in solving a problem meant to prove they have done some required work. Nodes who attempt to solve the problem are known as miners. These miners are always in a race to solve each problem as quickly as possible to build the next block. In return for their time and resources, they are paid transaction fees directly from the users making the transactions. They are also awarded by the network with Bitcoins explicitly created as a reward for mining the new block. These new coins that are created are the only way new coins are ever introduced to the system. Miners are trying to find the nonce for every new block. When creating a hash for a block, not just any value will work. The system requests a particular hash value that starts with a certain number of zeros. These extra constraints make the hash much more challenging to generate. Computers guess the nonce repeatedly until they finally come up with a value that gives us a hash that meets these constraints. Figure 20 presents a Proof of work mining diagram.

Furthermore, proof of work calculations requires a lot of computing power. All this computing comes at a cost both financially and to the environment. [47] Another problem in proof of work algorithm is the minor monopoly problem. Bitcoin miners that control a majority of the network dedicate more resources. As a result, they can validate the blocks according to their wish.

3.1.5 Proof of Stake

One very successful consensus algorithm is known as Proof of Stake. Proof of stake is a consensus algorithm that seeks to achieve consensus by giving a vote to those with the stake. In this case, stake means those who have the most investment in a positive outcome for this system. These users are considered the most likely to vote on making a positive impact on or improvement of the system.

In the proof of stake consensus protocol, Instead of miners, there are validators. These

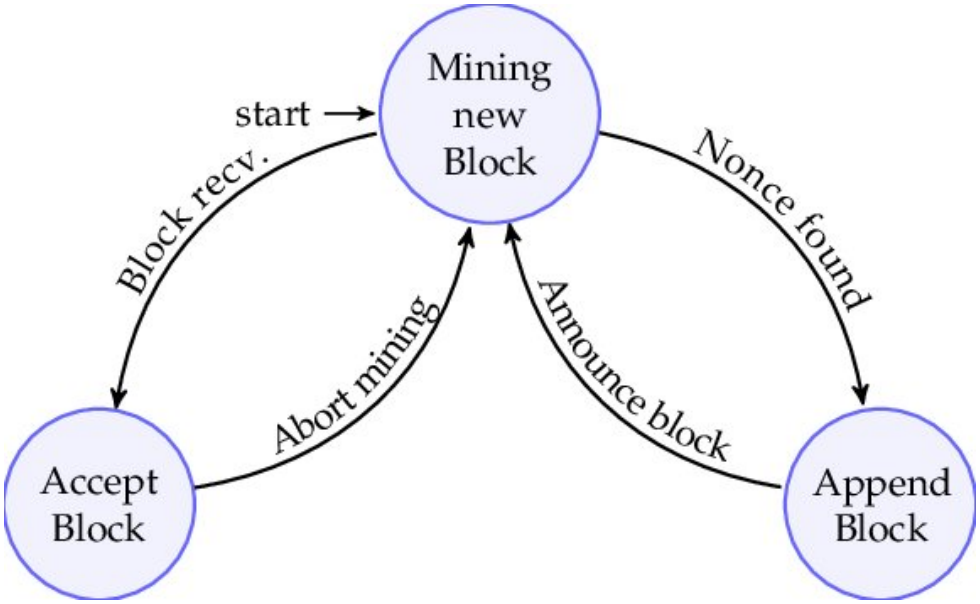

Figure 20: The state diagram of a miner in the original PoW.[39]

validators do not need to invest in computing equipment to mine the blocks that create coins because all the coins exist from the outset. Instead, the purpose of these validators, also known as stakeholders, is to determine which block makes it onto the blockchain. In order to validate transactions and create blocks, validators put up their coins as stake. If they validate a fraudulent transaction, they lose their holdings and their right to participate as a validator in the future. This check incentivizes the system to validate only truthful transactions. In proof of stake, the greater the fraction of the total coins a validator owns, the higher their chances of being picked to create the next block because they have a higher stake. If a validator's block is added, the validator will be awarded coins proportional to the amount of their bet. Figure 21 is illustrated consensus processes in proof of stake.

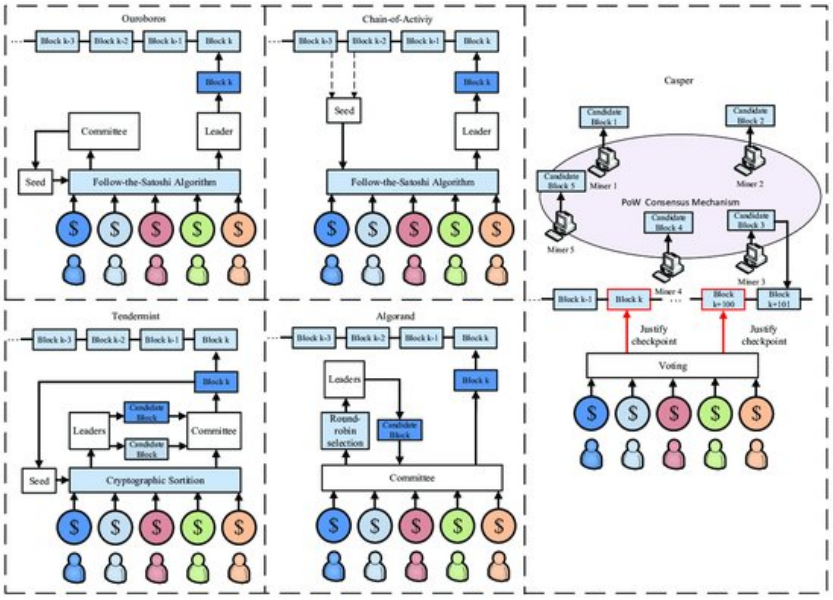

Figure 21: Illustrations of several PoS consensus processes.[58]

3.2 Types of Blockchain

There are three different types of blockchain, public, private, and consortium. The type of blockchain depends on the privacy of the information contained in blocks.

3.2.1 Public blockchain

The transactions can be verified and checked by all users. A public blockchain is decentralized. Bitcoin and Ethereum are public blockchain networks. [47] Every node in blockchain has access to public blockchain information.

3.2.2 Consortium blockchain

Consortium blockchain can be seen as partly decentralized, and it usually used for business to business transactions. The data that is stored in blocks can be public or private.

Hyperledger and R3CEV are consortium blockchains. [47]

3.2.3 Private blockchain

Data in a private blockchain is not accessible publicly but only from authorized users. [47] Private blockchain is protected from malicious users, and the identity of users that are doing the transactions is known.

3.3 Application of Blockchain

Blockchain technologies can be applied in the industry of finance but also other applications.

- *Cryptocurrency:* Bitcoin is the first implemented example of cryptocurrency, and it is an entirely digital currency. It is independent of government systems and banks. Bitcoin uses a method of encryption in order to transfer funds. Additionally, it managed public keys for bitcoin transactions. [47]

- *Smart Contract:* A smart contract is a computer program stored inside a blockchain.[33] They are immutable, and they are distributed. In a smart contract, users can store their data in all acceptable forms that any programming language permits.

- *Hyperledger:* Hyperledger is an open-source, collaborative effort created to advance cross-industry blockchain technologies. It is a global collaboration hosted by The Linux Foundation, including leaders in finance banking, Internet of Things, supply chains manufacturing, and technology. It is a framework to create, deploy, and maintain blockchains for businesses that ensure accountability, transparency, and trust among business partners. Hyperledger is a software used to create personalized blockchain service.

3.4 Blockchain and GDPR

The rapid development of cloud computing technology and the massive storage of users' data from online services erase the concern for a valuable legislation system that will protect the European Union citizens' data. Personal data is all the information that reveals the identity of a person directly or indirectly. These data include IP addresses, cookies, digital fingerprinting, and geolocation. [23] As a result, European Union introduced the General Data Protection Regulation(GDPR) on the 25th of May 2018, which offers users of online services the opportunity to control the provision of their data in several companies' databases. [74]

The GDPR has six general data protection principles: [23]

- *Fairness and lawfulness* principle requires that an organization collects personal data fairly and transparently. In addition, organizations have to inform the users about the exact reason their data is collected and the purpose. Furthermore, the data cannot be used against a person in any way.
- *Purpose limitation* which prevents organizations from collecting more personal data than needed for specific data processing.
- *Data minimization* determines how the data is processed. Personal data have to be used only for the processing mentioned in the agreement with users, and they cannot be used for other purposes.
- *Accuracy* requires personal data to be accurate and up to date. Differently, if data are incorrect or misleading, they must immediately be fixed or deleted.
- *Storage limitation* states that personal data can be saved only for as long as needed to fulfill the service that it was agreed. After the service or contract ends the data

must be removed or anonymized.

- *Integrity and confidentiality* obligates personal data is secure against hackers, unauthorized users, or accidental data leakage.

The decentralized nature of blockchain could be an effective solution to GDPR-compliant personal data management. [76]

3.4.1 Challenges

There are three law articles in the GDPR that are problematic in blockchain use. These are article 16 is about the right to rectification, article 17 about the right to be forgotten, and article 18 about the processing restriction.[77]

Article 16 gives the right to correct the data that someone has about an individual. The user has the right to change existing data that companies have about him, and he can also add new data if he claims that the current data is inaccurate or incomplete. The problem faced with blockchain from the above article is the difficulty of changing the data.[77]

Article 17 gives the right to be forgotten. Blockchain does not permit the removal of the data. As a result, EU citizens cannot exercise their right to delete their data. As a result, blockchains cannot comply with the GDPR. [77]

Moreover, Article 18 gives the right to prevent companies from doing something with the user's data. An exception to the article above is when the data is inaccurate or if it was illegally collected. This article's issue is that most blockchains are completely open, allowing anyone to take a copy of all the data stored and using it for any purpose. So there is no control over who is processing the data.[77] [13]

3.4.2 Solutions

There are several ways to solve the challenges presented above.

Firstly, the problem mentioned in articles 16 and 17 could be solved by storing personal data somewhere other than a blockchain like a secure server where there will be accessible only to read and write. Then a reference of the data will be stored on the blockchain like a shortcut or pointer.

On the other hand, this solution does not comply with blockchain's decentralized nature because an external server is used. This issue could be bypassed with zero-knowledge proof, which permits the transaction participants to prove the knowledge to each other without revealing any additional information about the knowledge. [45]

Finally, a solution for Article 18 of the GDPR would be to store the personal data in a permissioned blockchain instead of a public one. In contrast to public blockchains that allow anyone to see the data stored inside of them, permissioned blockchains provide a secure way for a group of entities to exchange information, know and identify each other when there is not fully trust between them. [5] [72]

3.5 Blockchain applications in pandemics for tracking purposes

One of the first examples of a contact tracing app with blockchain technology is presented in the Beeptrace protocol [82]. This approximation is based on the users' geolocation tracking after the user uploads the day's geodata, the server checks in the blockchain for possible contacts based on nodes' geolocation uploads.

Another solution is described in [49]. This approach suggests the update of the blockchain by a limited number of nodes. As they named in the research, several oracles are updating the blockchain with the condition of the patients. Finally, the application creates dashboards with graphs that describe the overall knowledge resulting from the blocks' information.

An also interesting approach to this issue has been proposed by [6] approach. In this

example, every user of the application is a node in the blockchain. Each infected user uploads everyday contacts. Thus, every node in the blockchain can check if it comes in contact with an infected individual.

4 Methodology

4.1 Ethereum technology and smart contracts

Ethereum is an open-source, programmable public blockchain platform. It includes its scripting language known as Solidity and has an entirely separate ecosystem of tools. The Etheruem Virtual Machine(EVM) is capable of executing logic, algorithm, and process data inputs. In an Ethereum environment, a user can create their operations. These operations can vary in complexity, which by design gives room for creating several decentralized applications, including cryptocurrencies and tokens. Moreover, a developer can write an application using a programming language call solidity modeled after Python, JavaScript, and submitted to an EVM for execution. ”Ether” is the cryptocurrency of Ethereum, and is used to pay transaction fees.

Ethereum accounts contain the following information:

- *The nonce*, a counter is used to ensure that there will not be duplicated transactions in the blockchain.
- *Account balance* which represents the number of ether.
- *Contract code*, a unique hash code for each contract
- *Storage* of the account which is empty in default.

Smart contracts on Etherium are programs that can be run on the Ethereum network. These programs are deployed to the entire network, and all participating nodes verify each call to a smart contract. This procedure allows for trusted parties to participate in complex financial contracts without any third-party supervision. Numerous projects are working

on prediction markets that allow anyone to bet on a future event and be paid out if their prediction was correct without any central authority. Other projects are working on loans, where anyone worldwide can request or offer funds for a loan entirely from a decentralized smart contract. These smart contracts will exist and be executable so long as the network exists.

Etherium, however, is the first blockchain to have implemented a smart contracts platform. In comparison to bitcoin scripts, Ethereum smart contracts are Turing Complete. The versatility for what is smart contracts can be used for has resulted in vastly different developer ecosystems compared to Bitcoin. The Ethereum developer ecosystem is rich in Turing and open-source projects. Many projects and entire companies emerged, with their core tech being built on Ethereum smart contracts.[10]

4.2 Technology

This experiment is using the Django framework. Django is a web development framework that is commonly used for information systems. It is open-source, and it uses python programming language in the backend. Additionally, for the deployment of a web contact tracing app in the framework, we used HTML5, CSS3, and javascript programming languages. [19]

Furthermore, for the blockchain part of the application, we used the Ethereum platform. The Solidity programming language is used for the development of the smart contract. The IDE used for developing the smart contract is named Remix, an open-source web and desktop application. In Remix, the user can test the execution correctness of the deployed contract. [40]

Besides, It is selected the truffle framework. Truffle is a growth, testing, and asset pipeline framework for Ethereum, which contains several features. One of the features

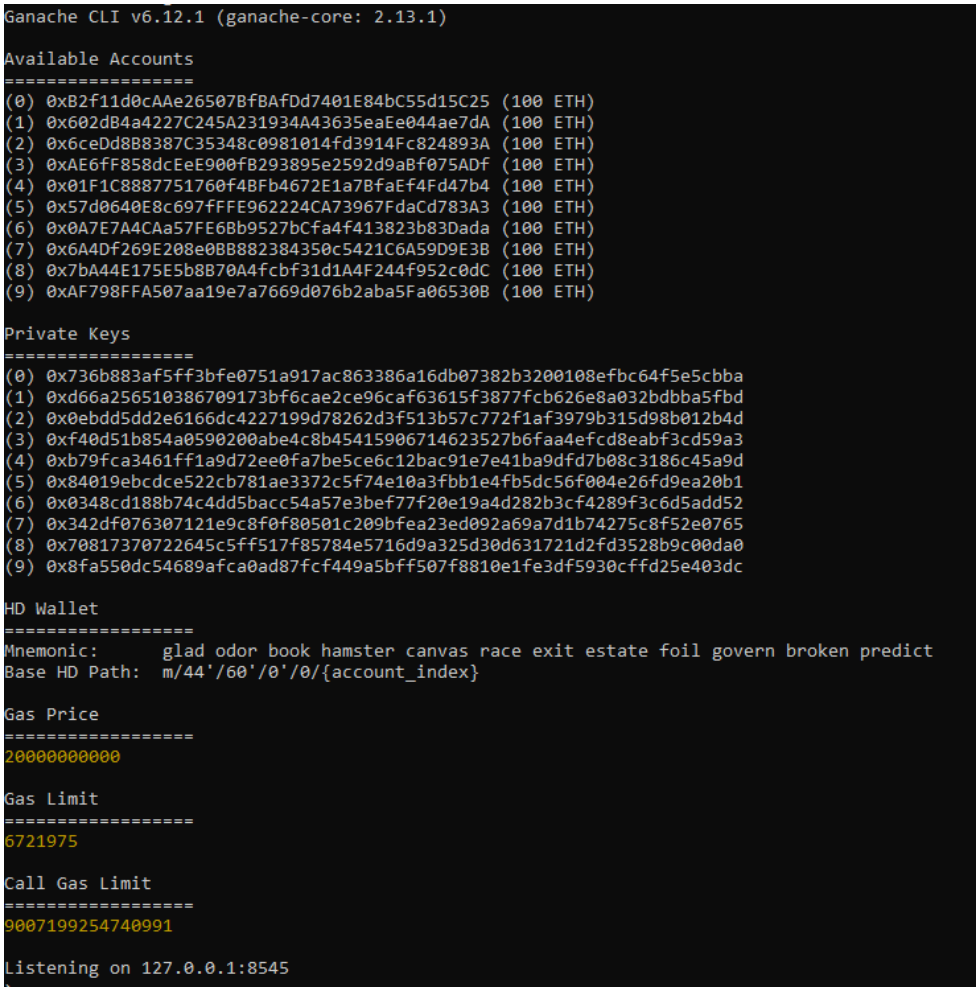

Figure 22: List of fake accounts in the terminal

is Ganache, which is used for fake account generation in the Ethereum platform. Every account has 100 ETH that is consumed in every transaction in the smart contract. Figure 22 displayed the hash code for every fake account in the Ganache framework and their private keys. It is also mentioned the amount of ETH in each account. [42]

4.3 Graphic representation of the experiment

The application of the experiment is designed according to hybrid architecture. A chosen centralized server and a decentralized blockchain in Ethereum is used. The tracing in the application is succeeded via the location of infected individuals.

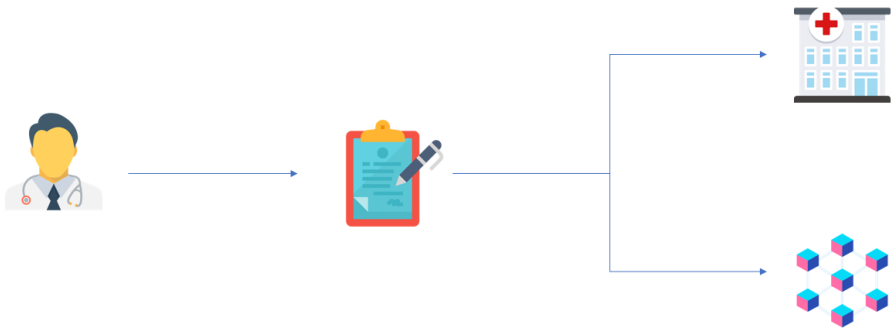

Figure 23: Health official view in application

In figure 23 is illustrated the first part of the use of the application. The health official completes a patient form with the patient's information and his condition. The patient's data are stored in the hospital's central server and blockchain.

The patient's data flow is presented in figure 24. According to this, all patients' data accompanied by doctor's notes are stored in a secure hospital database. At the same time, the postal code, country, the status of the patient (Infected, Suspected, Cured) are appended in a blockchain. Finally, to satisfy GDPR requirements, a hashing is generated and saved in both the central database as a primary key and blockchain as an id.

From a user perspective, the information streams are between the contact tracing web

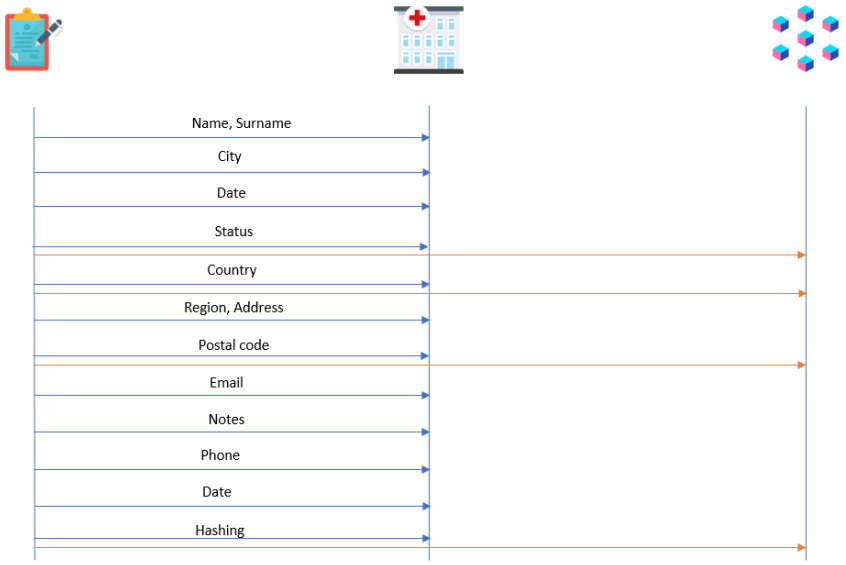

Figure 24: Store data in database and blockchain

application and the blockchain. The user searches about infected people in the area based on postal code, country, and city. The app requests the blockchain. Each user search interacts only with the public blockchain of Ethereum. There is no interaction with the hospital's database to avoid privacy issues. The process is shown in figure 25.

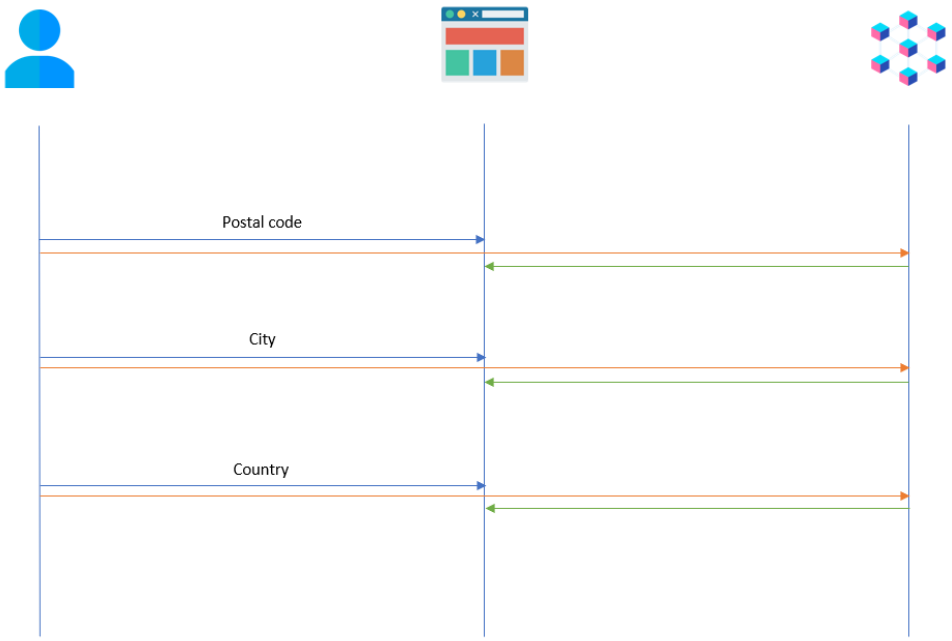

Figure 25: Retrieve data from blockchain

5 Experiment and deployment

The application's development started from the web application part where the data stored in the central database. Firstly, it is designed the user interface. The web application is designed for hospitals which are taking part in virus tracing. Also, there is a supplementary application that is used by individuals for the tracing of the infected people in the search area.

5.0.1 Application for hospitals

After the health official logged in to the website, the home page presented two options, the *Add Patients* button and the *Patients list* button. The same choices are shown in the navbar under the *Options* link, too. Figure 26 below shows the home page of the user.

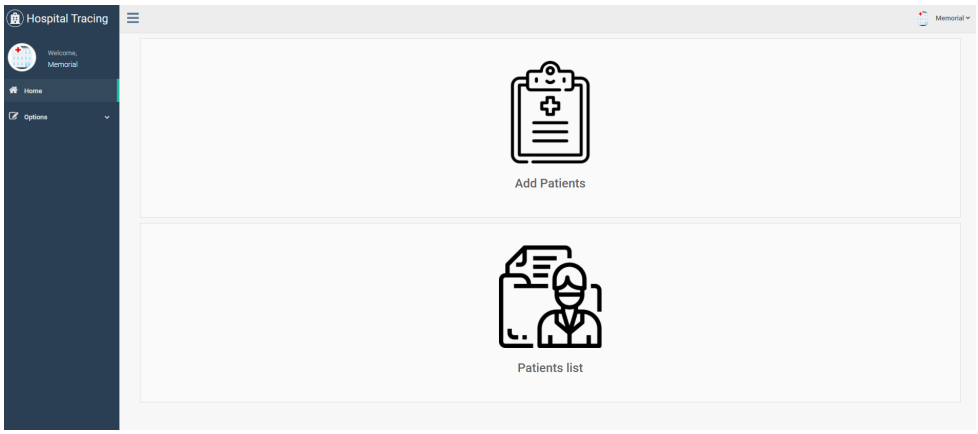

Figure 26: Home page

After the user clicks on the *Add Patients* button, the application leads him to the patient's form page, as is shown in figure 27 and 28. In the form, the health officer fills the name, the surname, the status (infected, suspected, cured), the location information

Figure 27: Patient's form part 1

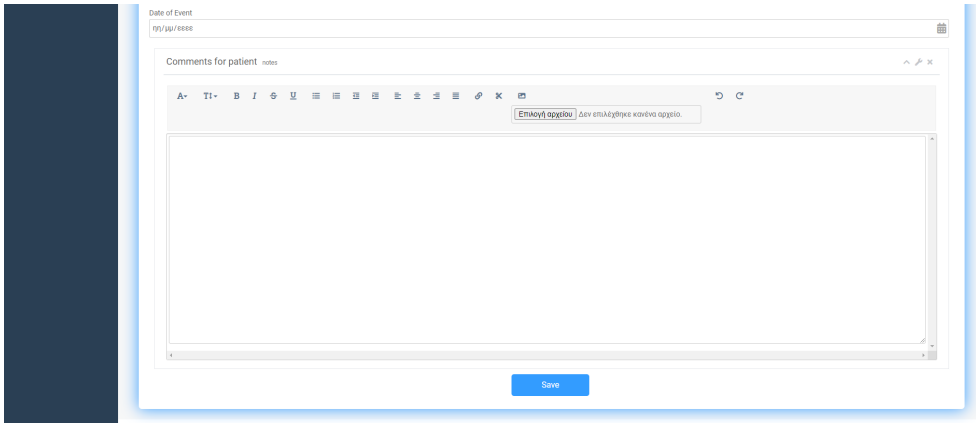

Figure 28: Patient's form part 2

(address, country, city, region, postal code), the phone number, the date, the email, and supplementary notes for the patient's condition.

The second option of the user is the *Patients list* button. The application request from the central server the number of patients that are stored in its database. Every hospital has its user. Sequentially, each user has access to its database and not to every patient stored in the blockchain.

Figure 29 illustrates the view of the patient's list. We can see that the list contains the

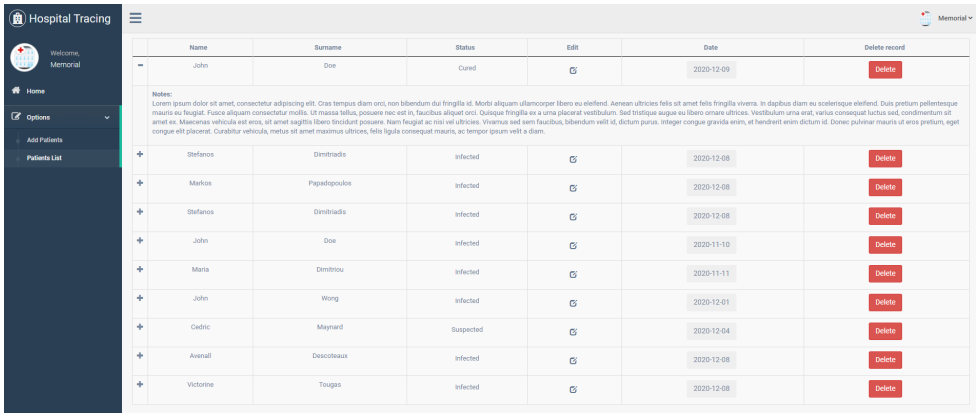

Figure 29: Patient's list

name, the surname, the status, the date, and the notes of each patient. The notes are shown after the user selects the plus symbol in the first column of the table. Furthermore, there are *edit* button and *delete* button. The functionality of the edit button updates the record of the user. The red delete button deletes the patient from the local database.

5.0.2 Application for tracing

The application which is used for the tracing part does not require login from the users. It is a simple search page where every individual could complete the city, the country, and the postal code and get infected people in the search area. Figure 30 displays the search page application. As mentioned above, the search page interacts only with the blockchain to calculate the search results.

5.1 Blockchain code analysis

The smart contract is developed in the Remix IDE Ethereum platform, as mentioned above. In listing 1, it is mentioned the assignment of the variables in the contract:

- *Statuses*: Enum is a type of variable that the user defines. We created Statuses as a

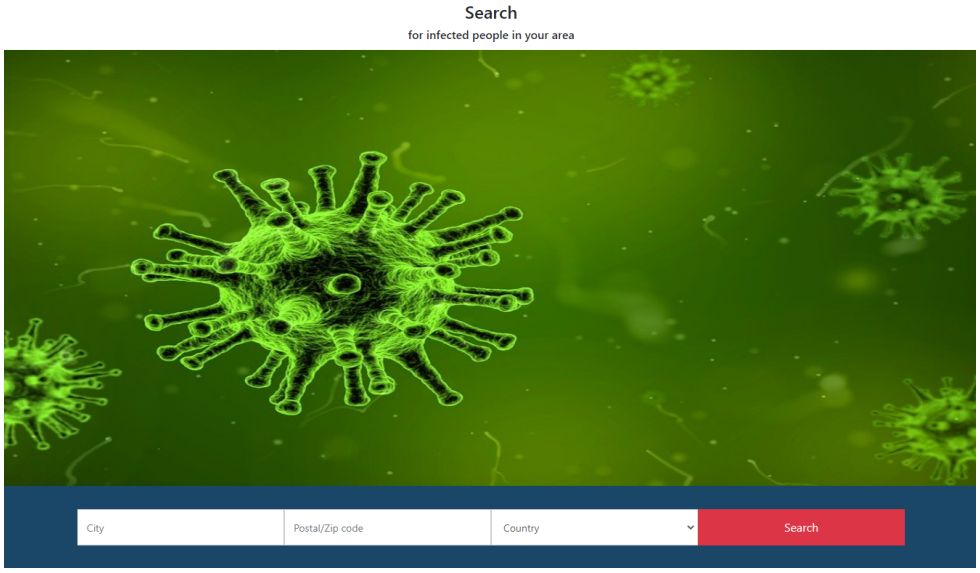

Figure 30: Search page of tracing app

variable that takes values zero, one, and two, performed as Infected, Suspected and Cured values.

- *currentStatus*: It is a Statuses type of variable that defined above.
- *Patient*: A struct is another custom type of variable defined by the user. Structs are a group of several associated properties determined by the same name. In our application, the Patient struct is determined by the variables below:
  - *postal*: The postal code of the patient's living area is assigned in a string variable.
  - *hospitalName*: In this string variable assigned the hospital name, the patient is diagnosed.
  - *hashing*: A string variable that represents the hashing generated by the patient's name and surname.

  - *country*: The country where the patient lives permanently is assigned in a string variable.
  - *status*: Statuses type of variable that assigned the condition of the patient. It takes values 0 for infected, 1 for Suspected, and 2 for Cured patients.

- *patients*: A list that saves each patient in the blockchain.

Listing 1: Solidity variables

```
enum Statuses { Infected, Suspected, Cured }
Statuses currentStatus;
Patient patient;



// Create a struct that saves the information of the infected person

struct Patient {
    Statuses status;
    string postal;
    string hospitalName;
    string hashing;
    string country;
}

Patient[] public patients;
```

The function ***addPatient*** that is specified by Solidity code in listing 2, it is used for adding patients in the list *patients* of the smart contract. The function adds Patient objects

in the list defined by status, postal, hospitalName, hashing, and country variables. The function returns the current number of elements in the list.

Listing 2: The function addPatient

```
function addPatient(Statuses _status, string memory _postal, string
    memory _hospitalName, string memory _hashing, string memory
    _country) public returns(uint) {

   patients.push(Patient(_status, _postal, _hospitalName, _hashing,
       _country));

   return patients.length;
}
```

The function ***getPatientsCount*** calculates the number of entries in the patient list. It is important to notify that the patient list is common for every hospital account in the application. Thus, the number returned each time the function is called is corresponding in entries of all hospitals.

Listing 3: The function getPatientsCount

```
function getPatientsCount() public view returns(uint) {
   return patients.length;
}
```

The last function of the contract is *getPatient*. It is a simple getter taken as an input the index of patients list and returns the corresponding Patient object. The code of the function is presented in listing 4.

Listing 4: The function getPatient

```
function getPatient(uint index) public view returns (Patient memory) {
   return patients[index];
}
```

5.2 Blockchain connection with python and Django framework

After developing the application user interface and smart contract, it is essential the connection of smart contract with the application. The app is constructed in python so, it is added to the source code the *Web3.py* library. The original API is in Javascript (Web3.js). The library is designed to interact with smart contracts for the execution of transactions. These transactions are achieved by calling the contract's functions in our main application. The functions that presented below are not complete. The entire python code presented in the Appendix section.

Listing 5 presents the encryption function that generates the hashing from the user's name and surname. It is used the encoding algorithm SHA 256, which is the algorithm that is utilized within the Bitcoin network for mining and creation of Bitcoin addresses. The function takes as an input a string value, and it returns a hashing string value.

Listing 5: The encrypton function

```
def encrypt_string(hash_string):
   sha_signature = hashlib.sha256(hash_string.encode()).hexdigest()
   return sha_signature
```

Firstly, the application has to connect with the Ethereum blockchain network and smart contract. For the connection, we used the Ganache framework that creates fake Ethereum

accounts for testing. The web3 API connects to the HTTP provider that, in our case, is the localhost. In addition, in ABI variable is assigned the contract's Application Binary Interface that is resulting after the contract is compiled. Furthermore, the address of the contract is needed to complete the connection. This value is assigned in the address string variable. The contract variable stores the connection. The implementation with the python code is presented in listing 6.

Listing 6: The connection of smart contract with python application

```
ganache_url = 'http://127.0.0.1:8545'
web3 = Web3(Web3.HTTPProvider(ganache_url))
abi = json.loads(JSON format for a 'contracts interface)
address = "0x######################"
contract = web3.eth.contract(address = address, abi = abi)
```

Next, the function *add_patients* in the web application is used to add patients to the hospital's central server and blockchain. The function with web3 API calls the *addPatient* function of the smart contract. The blockchain transaction seceded by added the public key of the Ethereum account, which is a hashing code. In listing 7 is shown only the python code that makes the connection to the blockchain.

Listing 7: The Web3 use in adding patients function

```
def add_patients(request):


      result = contract.functions.addPatient(status, postal,
          str(precord.user), encrypt_string(name_surname),
          country).transact({'from':'0x###############'})
```

Similarly, the *delete_patient* function update the blockchain. As is already mentioned

above, the blockchain is immutable. If one of the blocks is deleted, the blockchain is becoming invalid. Sequentially, deletion of a patient is not possible in the blockchain. For this purpose, the function updates the blockchain by using *addPatient*, and it takes in the hospital name input the word ”delete”. It is using as identification of the hashing value that is generated by the patient’s name and surname. Listing 8 contains only the python code for the web3 API connection.

Listing 8: The the Web3 use in deleting patients function

```
def delete_patient(request, uid):


   result = contract.functions.addPatient(status, postal, 'deleted',
       str(hashing), country).transact({'from':'0x###############'})
```

Listing 9 describes the code of *update_patient_status* function. It is acts like the *delete_patient* function. The only difference is in the the inputs which are the same as *add_patients* function.

Listing 9: The the Web3 use in updating patients function

```
def update_patient_status(request, uid):


   result = contract.functions.addPatient(status, postal,
       str(patient.user), str(patient.hashing),
       country).transact({'from':'0x###############'})
```

Lastly, it is developed a supplementary application for individuals who search for infected people in the area. The function that is used for this purpose is named *search_results*. In the function is called the *getPatientsCount* and the result is assigned in the variable no_patients. Subsequently, the number no_patients is used to set the range of iterations

needed to find the infected individuals. The iteration takes place in the blockchain via *get-Patient* function of smart contract. The result is saved in the list lpatient. In the list, lhash is stored the unique, valid hash values of the infected people. The number is saved on a counter. The full code is presented in listing 10.

Listing 10: The Web3 use in search function infected people in your area

```
def search_results(request):


   no_patients = contract.functions.getPatientsCount().call()


   lpatient = []
   lhash = []
   message = 'No infected people'


   counter = 0
   for i in range(0, no_patients):
      patients = contract.functions.getPatient(i).call()
      lpatient = list(patients)
      if lpatient[0] == 0 and lpatient[1] == postal and lpatient[4]
         == country:
         if lpatient[3] not in lhash and lpatient[2] != 'deleted':
            lhash.append(lpatient[3])
            counter = counter + 1


         message = 'Infected people in your area are'
```

6 Results and discussion

Nodes add the blocks in blockchains. More specifically, the blockchain cannot be created by users that not be part of the distributed network. As a result, there are two options for the development of the blockchain contact tracing app. The application could be produced as user-centered. Thus, every user must be part of the Ethereum network to use the app. The user also needs to have an amount of Ethereum in the digital wallet to pay for each transaction made by the smart contract. The second option is more optimal for our experiment, and it needs fewer resources because it is based on fewer Ethereum accounts that will add blocks to the blockchain. Furthermore, users do not charge. More specifically, every hospital is a node of the blockchain network, and it performs all transactions by their accounts.

Furthermore, this experiment used blockchain to leverage the pseudonymity that blockchain technology provides. Consequently, the final application will be secure and GDPR regulated because the patients could not identify the information stored in the public blockchain. As mentioned above, the nodes in this blockchain are each hospital that uses the web app. Sequentially, the patient's pseudonymity is provided by the application before the information is uploaded to the blockchain. The patient's name and surname generate this hashing. It is stored both in the blockchain and central server of each hospital.

As mentioned in the Methodology section, the delete and update commands could not be used in the immutable blockchain technology. These procedures are accomplished by updating the blockchain with the delete information that defines the deletion and changing the information with the same hash code to identify the same patient.

Moreover, the GDPR challenges are faced effectively by the experiment's application. The use of central secure hospitals' servers that save the data and are accessed only to read

and write is the optimal solution proposed above for GDPR articles 16 and 17. Additionally, the blockchain is using the hashing as a pointer to this data. On the other hand, the application's design failed to comply with the regulation in article 18. Because of the app's decentralized nature, data are public, and everyone can have a copy. The most significant part of GDPR is the anonymity of the data. In the application, the identity of the users is stored in the secure hospital's server and the data that are published in the blockchain are anonymous by the use of hashing that the application generates.

Finally, it is essential to compare the already existing blockchain technology contact tracing approaches with this experiment. In contrast to the existing applications designed by the decentralized approach, the experiment application is created based on a hybrid architecture. Thus, the application is GDPR regulated. Furthermore, the experiment does not use geodata or contact data from individuals that used the app.

7 Conclusion

A blockchain technology approach is proposed to solve the GDPR legislation issues in contact with tracing apps for pandemics. Introduced the Blockchain technology and its use. Described the architecture of the existing contact tracing applications and their functionality. Detailed the methods that are followed and the technologies that are used for the development. Explained challenges that are faced during the design of the application and they the optimal solutions are discussed. The differences in the application with the already published approaches are presented. In conclusion, the application is tested for its compliance with the GDPR legislation and strengths and weaknesses of the application are mentioned.

7.1 Benefits and limitations

During this project, several approaches are studied on the possible solutions to the problem. The analysis of the existing solutions in contact tracing developed with and without blockchain technology gave a clear view of the final application structure.

The application's hybrid architecture could solve more than half of GDPR issues in blockchain technology. The application compounded a hospital information system with a contact tracing app approach without arising privacy issues to the data. With this solution is avoided the falsely spread of the data to unauthorized users.

On the other hand, there are limitations to the solution that is proposed. It is already noted that the application cannot solve the problem faced in the GDPR about the users asked for permanent deletion from the blockchain. Additionally, the application is not tested for the user experienced, and it is not designed in detail its user interface because of limited time. Another issue is that the security of the central server in hospitals is taken as

a prerequisite. This issue has to be analyzed in more depth before installing the application and has to be covered with any cybersecurity issues.

7.2 Recommendations for future work

This study succeeded in developing a different approach to blockchain contact tracing an application using a hybrid architecture. However, some aspects did not take into account during the process because of the lack of time.

An interesting approach for further investigation could be a complete decentralized application based only on a private blockchain. There were no privacy issues in this specter, and there is no need to store data in a centralized database.

Moreover, suppose blockchain technology is adopted for contact tracing in EU countries. In that case, it is essential to solve the problem presented in the GDPR article about every individual's right within the EU to be forgotten. So, it has to study the potential solution with the use of a permissioned blockchain that is mentioned in section 4.

Finally, the application could be developed in a mobile version and be combined with other technologies such as real-time geolocation and Bluetooth, which could enrich the app's existing functionalities.

Bibliography

[1] Susu A. *COVID-19: Fiji acquires contact tracing software*. https://www.fijitimes.com/covid-19-fiji-acquires-contact-tracing-software/. Accessed: 2020-11-29.

[2] Dotun Adebanjo et al. "Managing and Recovering from the COVID-19 Pandemic– Dubai We Learn Research Report". In: (2020).

[3] Nadeem Ahmed et al. "A survey of covid-19 contact tracing apps". In: *IEEE Access* 8 (2020), pp. 134577–134601.

[4] Julia Amann, Joanna Sleigh, and Effy Vayena. "Digital contact-tracing during the Covid-19 pandemic: an analysis of newspaper coverage in Germany, Austria, and Switzerland". In: *medRxiv* (2020).

[5] Elli Androulaki et al. "Hyperledger fabric: a distributed operating system for permissioned blockchains". In: *Proceedings of the thirteenth EuroSys conference*. 2018, pp. 1–15.

[6] Md Murshedul Arifeen et al. "Blockchain-enable contact tracing for preserving user privacy during COVID-19 outbreak". In: (2020).

[7] Muhammad Ajmal Azad et al. "A First Look at Privacy Analysis of COVID-19 Contact Tracing Mobile Applications". In: *IEEE Internet of Things Journal* (2020).

[8] Jason Bay et al. "BlueTrace: A privacy-preserving protocol for community-driven contact tracing across borders". In: *Government Technology Agency-Singapore, Tech. Rep* (2020).

[9] Rachel Butcher and Norman Fenton. "Extending the range of symptoms in a Bayesian Network for the Predictive Diagnosis of COVID-19". In: *medRxiv* (2020).

[10] Vitalik Buterin et al. “A next-generation smart contract and decentralized application platform”. In: *white paper* 3.37 (2014).

[11] Claude Castelluccia et al. “DESIRE: A Third Way for a European Exposure Notification System Leveraging the best of centralized and decentralized systems”. In: (2020).

[12] Hyunghoon Cho, Daphne Ippolito, and Yun William Yu. “Contact tracing mobile apps for COVID-19: Privacy considerations and related trade-offs”. In: *arXiv preprint arXiv:2003.11511* (2020).

[13] Fábio Coelho and George Younes. “The GDPR-Blockchain paradox: A work around”. In: *Conference: 1st workshop on GDPR compliant systems, co-located with 19th ACM international middleware conference*. 2018.

[14] *Contact tracing apps in Indonesia*. `https://www.nortonrosefulbright.com/-/media/files/nrf/nrfweb/contact-tracing/indonesia-contact-tracing.pdf?revision=1c30d2b8-e883-4878-beee-f6fc5a6eb7eb&la=en-ca`. Accessed: 2020-11-22.

[15] Ginbert Cuaton, Las Johansen Caluza, and Joshua Francisco Neo. “COVID-19 Health Response from January to April 2020 in the Philippines: A Topic Modeling Analysis using Latent Dirichlet Allocation Algorithm”. In: *Available at SSRN 3590910* (2020).

[16] Aaqib Bashir Dar et al. “Applicability of mobile contact tracing in fighting pandemic (covid-19): Issues, challenges and solutions”. In: *Computer Science Review* (2020), p. 100307.

[17] Paul-Olivier Dehaye and Joel Reardon. “SwissCovid: a critical analysis of risk assessment by Swiss authorities”. In: *arXiv preprint arXiv:2006.10719* (2020).

[18] Whitfield Diffie and Martin Hellman. “New directions in cryptography”. In: *IEEE transactions on Information Theory* 22.6 (1976), pp. 644–654.

[19] *Django framework*. `https://www.djangoproject.com/`. Accessed: 2020-11-29.

[20] Ali Dorri et al. “LSB: A Lightweight Scalable Blockchain for IoT security and anonymity”. In: *Journal of Parallel and Distributed Computing* 134 (2019), pp. 180–197.

[21] Weidong Fang et al. “Digital signature scheme for information non-repudiation in blockchain: a state of the art review”. In: *EURASIP Journal on Wireless Communications and Networking* 2020.1 (2020), pp. 1–15.

[22] Michael J Fischer. “The consensus problem in unreliable distributed systems (a brief survey)”. In: *International conference on fundamentals of computation theory*. Springer. 1983, pp. 127–140.

[23] Michelle Goddard. “The EU General Data Protection Regulation (GDPR): European regulation that has a global impact”. In: *International Journal of Market Research* 59.6 (2017), pp. 703–705.

[24] Rajan Gupta et al. “Analysis of COVID-19 Tracking Tool in India: Case Study of Aarogya Setu Mobile Application”. In: *Digital Government: Research and Practice* 1.4 (2020), pp. 1–8.

[25] Shima Hamidi, Sadegh Sabouri, and Reid Ewing. “Does density aggravate the COVID-19 pandemic? Early findings and lessons for planners”. In: *Journal of the American Planning Association* (2020), pp. 1–15.

[26] Marwah Hassounah, Hafsa Raheel, and Mohammed Alhefzi. “Digital Response During the COVID-19 Pandemic in Saudi Arabia”. In: *Journal of Medical Internet Research* 22.9 (2020), e19338.

[27] Czech Ministry of Health. *eRouka app: Fighting against COVID-19 via privacy-first Bluetooth tracing*. https://erouska.cz/. Accessed: 2020-11-29.

[28] Czech Ministry of Health. *Thai Covid-19 apps judged invasive*. https://www.bangkokpost.com/business/1954287/thai-covid-19-apps-judged-invasive. Accessed: 2020-11-29.

[29] Csilla Herendy. "How were apps developed during, and for, COVID-19?: An investigation into user needs assessment and testing". In: *2020 11th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*. IEEE. 2020, pp. 000503–000508.

[30] Bronwyn E Howell and Petrus H Potgieter. "A Tale of Two Contact-Tracing Apps–Comparing Australia's COVIDSafe and New Zealand's NZ COVID Tracer". In: *Available at SSRN 3612596* (2020).

[31] Bobbie Johnson. *Nearly 40% of Icelanders are using a covid app—and it hasn't helped much*. https://www.technologyreview.com/2020/05/11/1001541/iceland-rakning-c19-covid-contact-tracing. Accessed: 2020-11-29.

[32] Fahrettin Koca. "Turkey's Management of COVID-19: Measures and Strategies of Health Policies". In: *Insight Turkey* 22.3 (2020), pp. 55–66.

[33] Ahmed Kosba et al. "Hawk: The blockchain model of cryptography and privacy-preserving smart contracts". In: *2016 IEEE symposium on security and privacy (SP)*. IEEE. 2016, pp. 839–858.

[34] Emmanouil Koulas. "Defining sovereignty and national interest on cyberspace: national and supranational paradigms". In: (2019).

[35] Vasileios Kouliaridis, Georgios Kambourakis, and Dimitrios Geneiatakis. “Dissecting contact tracing apps in the Android platform”. In: *arXiv preprint arXiv:2008.00214* (2020).

[36] Danilo Krivokapić, Bojan Perkov, and Davor Marko. “State of pandemonium: Digital rights in the Western Balkans and COVID-19”. In: (2020).

[37] Junko Kurita, Tamie Sugawara, and Yasushi Ohkusa. “Effectiveness of COCOA, a COVID- 19 contact notification application, in Japan”. In: *medRxiv* (2020).

[38] Samuel Lalmuanawma, Jamal Hussain, and Lalrinfela Chhakchhuak. “Applications of machine learning and artificial intelligence for Covid-19 (SARS-CoV-2) pandemic: A review”. In: *Chaos, Solitons & Fractals* (2020), p. 110059.

[39] Noureddine Lasla et al. “Green-PoW: An Energy-Efficient Blockchain Proof-of-Work Consensus Algorithm”. In: *arXiv preprint arXiv:2007.04086* (2020).

[40] Rana M Amir Latif et al. “A remix IDE: smart contract-based framework for the healthcare sector by using Blockchain technology”. In: *Multimedia Tools and Applications* (2020), pp. 1–24.

[41] Terence Lee and Howard Lee. “Tracing surveillance and auto-regulation in Singapore:‘smart’responses to COVID-19”. In: *Media International Australia* 177.1 (2020), pp. 47–60.

[42] Wei-Meng Lee. “Testing Smart Contracts Using Ganache”. In: *Beginning Ethereum Smart Contracts Programming*. Springer, 2019, pp. 147–167.

[43] Douglas J Leith and Stephen Farrell. “Contact tracing app privacy: What data is shared by Europe’s GAEN contact tracing apps”. In: *Testing Apps for COVID-19 Tracing (TACT)* (2020).

[44] Mitch Leslie. “COVID-19 Fight enlists digital technology: Contact tracing apps”. In: *Engineering (Beijing, China)* (2020).

[45] Wanxin Li et al. “Privacy-preserving traffic management: A blockchain and zero-knowledge proof inspired approach”. In: *IEEE Access* 8 (2020), pp. 181733–181743.

[46] Fan Liang. “COVID-19 and Health Code: How Digital Platforms Tackle the Pandemic in China”. In: *Social Media+ Society* 6.3 (2020), p. 2056305120947657.

[47] Iuon-Chang Lin and Tzu-Chun Liao. “A survey of blockchain security issues and challenges.” In: *IJ Network Security* 19.5 (2017), pp. 653–659.

[48] Peter Ó Máille. “The Anarchist Library Anti-Copyright”. In: (2020).

[49] Dounia Marbouh et al. “Blockchain for COVID-19: Review, Opportunities, and a Trusted Tracking System”. In: *Arabian Journal for Science and Engineering* (2020), pp. 1–17.

[50] Floris van der Marel. “Tracing for the people”. In: *Human-Centred Research and Design in Crisis* (2020).

[51] Tania Martin et al. “Demystifying COVID-19 digital contact tracing: A survey on frameworks and mobile apps”. In: *Wireless Communications and Mobile Computing* 2020 (2020).

[52] Washington Post-University of Maryland. *Most americans are not willing or able use an app tracking coronavirus infections thats problem big techs plan slow pandemic*. `https://www.washingtonpost.com/technology/2020/04/29/most-americans-are-not-willing-or-able-use-an-app-tracking-coronavirus-infections-thats-problem-big-techs-plan-slow-pandemic/`. Accessed: 2020-10-24.

[53] Ilaria Montagni et al. “The French Covid-19 contact tracing app: usage and opinions by students in the health domain”. In: *medRxiv* (2020).

[54] Johannes Müller, Mirjam Kretzschmar, and Klaus Dietz. “Contact tracing in stochastic and deterministic epidemic models”. In: *Mathematical biosciences* 164.1 (2000), pp. 39–64.

[55] Satoshi Nakamoto. “Bitcoin: A peer-to-peer electronic cash system Bitcoin: A Peer-to-Peer Electronic Cash System”. In: *Bitcoin. org. Disponible en https://bitcoin. org/en/bitcoin-paper* (2009).

[56] Anvar Narzullaev, Zahriddin Muminov, and Mavlutdin Narzullaev. “Contact Tracing of Infectious Diseases Using Wi-Fi Signals and Machine Learning Classification”. In: *2020 IEEE 2nd International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*. IEEE. 2020, pp. 1–5.

[57] Umaer Naseer et al. “Use of the Smittestopp app for contact tracing: validation study protocol”. In: (2020).

[58] Cong T Nguyen et al. “Proof-of-stake consensus mechanisms for future blockchain networks: fundamentals, applications and opportunities”. In: *IEEE Access* 7 (2019), pp. 85727–85745.

[59] Michael Nofer et al. “Blockchain”. In: *Business & Information Systems Engineering* 59.3 (2017), pp. 183–187.

[60] D. Normile. *Coronavirus cases have dropped sharply in South Korea. Whats the secret to its success?* `https://www.sciencemag.org/news/2020/03/coronavirus-cases-have-dropped-sharply-south-korea-whats-secret-its-success`. Accessed: 2020-10-24.

[61] Patrick Howell O'Neillarchive. *No, coronavirus apps don't need 60% adoption to be effective*. https://www.technologyreview.com/2020/06/05/1002775/covid-apps-effective-at-less-than-60-percent-download. Accessed: 2020-10-24.

[62] PH O'Neill, T Ryan-Mosley, and B Johnson. *A flood of coronavirus apps are tracking us. Now it's time to keep track of them*. 2020.

[63] Stamatis Papangelou and Sofia Papadaki. "Digital Currencies: A Multivariate GARCH Approach". In: *Mathematical Research for Blockchain Economy*. Springer, 2020, pp. 61–75.

[64] Olivier Pereira. *Why Should We Install the Coronalert Contact Tracing App?* Tech. rep. 2020.

[65] Benny Pinkas and Eyal Ronen. *Hashomer-a proposal for a privacy-preserving bluetooth based contact tracing scheme for hamagen*. 2020.

[66] Megha Rajagopalan. *Qatar Has Made Its Coronavirus Contact Tracing App Mandatory*. https://www.buzzfeednews.com/article/meghara/qatar-coronavirus-tracking-app-mandatory. Accessed: 2020-11-29.

[67] Abinaya Megan Ramakrishnan et al. "From Symptom Tracking to Contact Tracing: A Framework to Explore and Assess COVID-19 Apps". In: *Future Internet* 12.9 (2020), p. 153.

[68] Jens Helge Reelfs, Oliver Hohlfeld, and Ingmar Poese. "Corona-Warn-App: Tracing the Start of the Official COVID-19 Exposure Notification App for Germany". In: *arXiv preprint arXiv:2008.07370* (2020).

[69] Ronald L Rivest et al. *Pact: Private automated contact tracing*. 2020.

[70] Ian Anderson Rosie Garthwaite. *Coronavirus: Alarm over 'invasive' Kuwait and Bahrain contact-tracing apps*. https://www.bbc.com/news/world-middle-east-53052395. Accessed: 2020-11-29.

[71] Anthony Kwabena Sarfo and Shankar Karuppannan. "Application of geospatial technologies in the covid-19 fight of Ghana". In: *Transactions of the Indian National Academy of Engineering* 5.2 (2020), pp. 193–204.

[72] Harish Sukhwani et al. "Performance modeling of PBFT consensus process for permissioned blockchain network (hyperledger fabric)". In: *2017 IEEE 36th Symposium on Reliable Distributed Systems (SRDS)*. IEEE. 2017, pp. 253–255.

[73] Risala Tasin Khan Tanvir Rahman Taslima Ferdaus Shuva. "A Review of Contact Tracing Approaches for Controlling COVID-19 Pandemic". In: 2020 (2020).

[74] Christina Tikkinen-Piri, Anna Rohunen, and Jouni Markkula. "EU General Data Protection Regulation: Changes and implications for personal data collecting companies". In: *Computer Law & Security Review* 34.1 (2018), pp. 134–153.

[75] Thi Phuong Thao Tran et al. "Rapid response to the COVID-19 pandemic: Vietnam government's experience and preliminary success". In: *Journal of Global Health* 10.2 (2020).

[76] Nguyen Binh Truong et al. "Gdpr-compliant personal data management: A blockchain-based solution". In: *IEEE Transactions on Information Forensics and Security* 15 (2019), pp. 1746–1761.

[77] EUROPEAN UNION. *General Data Protection Regulation*. https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679. Accessed: 2020-10-25.

[78] David Watts. “COVIDSafe, Australia’s Digital Contact Tracing App: The Legal Issues”. In: *Australia’s Digital Contact Tracing App: The Legal Issues (May 2, 2020)* (2020).

[79] *What Ever Happened to Digital Contact Tracing?* `https://www.lawfareblog.com/what-ever-happened-digital-contact-tracing`. Accessed: 2020-11-29.

[80] Jacqui Wise. *Covid-19: UK drops its own contact tracing app to switch to Apple and Google model*. 2020.

[81] Stephen Woodman. “Hackers paradise: Hackers across Latin America are taking advantage of the current crisis to access people’s personal data. If not protected it could spell disaster”. In: *Index on Censorship* 49.2 (2020), pp. 40–42.

[82] Hao Xu et al. “BeepTrace: Blockchain-enabled Privacy-preserving Contact Tracing for COVID-19 Pandemic and Beyond”. In: *arXiv preprint arXiv:2005.10103* (2020).

[83] Zibin Zheng et al. “An overview of blockchain technology: Architecture, consensus, and future trends”. In: *2017 IEEE international congress on big data (BigData congress)*. IEEE. 2017, pp. 557–564.

[84] Zibin Zheng et al. “Blockchain challenges and opportunities: A survey”. In: *International Journal of Web and Grid Services* 14.4 (2018), pp. 352–375.

A Appendix

Listing 11: Python class views.py for hospital application

```
from django.shortcuts import render
from django.template import loader
from django.http import HttpResponse
from django.shortcuts import render, redirect
from .models import Patient
from django.contrib import messages
import pandas as pd
from django.contrib.auth.decorators import login_required
from web3 import Web3
import datetime
import hashlib
import json


def encrypt_string(hash_string):
    sha_signature = \
        hashlib.sha256(hash_string.encode()).hexdigest()
    return sha_signature


@login_required()
def index(request):
    context = {}
    template = loader.get_template('index.html')
    return HttpResponse(template.render(context, request))
```

```
@login_required()
def patient_form(request):

    dt = pd.read_csv('contact/static/info/countries.txt', sep='\n')
    countries = []
    df = dt.to_dict()
    for k, country in df.items():
        for k,v in country.items():
            countries.append(v)

    return render(request, 'patient_form.html', {'countries':
        countries})



@login_required()
def add_patients(request):

    ganache_url = 'http://127.0.0.1:7545'

    web3 = Web3(Web3.HTTPProvider(ganache_url))

    abi = json.loads('[{JSON format for a 'contracts interface}]')
```

```
address = "0xa84580e93474b942b48B16CAEeaA1920962CBd90"

contract = web3.eth.contract(address = address, abi = abi)

print(contract)

if request.method == 'GET':

    precord = Patient()

    precord.name = request.GET.get("name")
    precord.surname = request.GET.get("surname")
    precord.address = request.GET.get("address")
    precord.email = request.GET.get("email")
    precord.city = request.GET.get("city")
    precord.region = request.GET.get("region")
    precord.postal = request.GET.get("postal")
    precord.country = request.GET.get("country")
    precord.phone = request.GET.get("phone")
    precord.status = request.GET.get("status")
    precord.notes = request.GET.get("notes", None)
    precord.created_at = request.GET.get("bdate")
    precord.user = request.user

    name_surname = precord.name+"_"+precord.surname
    precord.hashing = encrypt_string(name_surname)
    country = precord.country
```

```
        postal = precord.postal
        status = int(precord.status)

        result = contract.functions.addPatient(status, postal,
            str(precord.user), encrypt_string(name_surname),
            country).transact({'from':'0x######'})
        num = contract.functions.getPatientsCount().call()
        print(num)

        precord.save()

        messages.success(request, 'Record Saved')

        return render(request, 'index.html')
    else:
        return render(request, 'index.html')

@login_required()
def patients_list(request):

    patients = Patient.objects.all()

    return render(request, 'patient_list.html', {'patients':patients})



@login_required
def delete_patient(request, uid):
```

```
ganache_url = 'http://127.0.0.1:7545'

web3 = Web3(Web3.HTTPProvider(ganache_url))

abi = json.loads('[{JSON format for a 'contracts interface}]')

address = "0xa84580e93474b942b48B16CAEeaA1920962CBd90"

contract = web3.eth.contract(address = address, abi = abi)

patient = Patient.objects.get(uid=uid)
patient.status = request.GET.get("status")
patient.notes = request.GET.get("notes", None)
patient.created_at = request.GET.get("bdate")

hashing = patient.hashing
country = patient.country
postal = patient.postal
status = int(patient.status)

result = contract.functions.addPatient(status, postal, 'deleted',
    str(hashing), country).transact({'from':'0x###############'})
patient.delete()

return redirect("http://127.0.0.1:8000/patients-list")
```

```
@login_required
def update_patient_status(request, uid):

    ganache_url = 'http://127.0.0.1:7545'

    web3 = Web3(Web3.HTTPProvider(ganache_url))

    abi = json.loads('[{JSON format for a 'contracts interface}]')

    address = "0xa84580e93474b942b48B16CAEeaA1920962CBd90"

    contract = web3.eth.contract(address = address, abi = abi)

    if request.method == 'GET':

        patient = Patient.objects.get(uid=uid)
        patient.status = request.GET.get("status")
        patient.notes = request.GET.get("notes", None)
        patient.created_at = request.GET.get("bdate")



        hashing = patient.hashing
        print(patient)
        country = patient.country
        postal = patient.postal
        status = int(patient.status)
```

```
        result = contract.functions.addPatient(status, postal,
            str(patient.user), str(patient.hashing),
            country).transact({'from':'0x############'})

        patient.save()

        return redirect("http://127.0.0.1:8000/patients-list")

    return redirect("http://127.0.0.1:8000/patients-list")


@login_required
def edit(request, uid):
    patient = Patient.objects.get(uid=uid)
    return render(request,'edit.html', {'patient':patient})


def users(request):

    dt = pd.read_csv('contact/static/info/countries.txt', sep='\n')
    countries = []
    df = dt.to_dict()
    for k, country in df.items():
        for k,v in country.items():
            countries.append(v)
```

```
    return render(request,'users.html', {'countries':countries})
```

Listing 12: Python class views.py for users application

```
def search_results(request):

    dt = pd.read_csv('contact/static/info/countries.txt', sep='\n')
    countries = []
    df = dt.to_dict()
    for k, country in df.items():
        for k,v in country.items():
            countries.append(v)

    ganache_url = 'http://127.0.0.1:7545'

    web3 = Web3(Web3.HTTPProvider(ganache_url))

    abi = json.loads('[{JSON format for a 'contracts interface}]')

    address = "0xa84580e93474b942b48B16CAEeaA1920962CBd90"

    contract = web3.eth.contract(address = address, abi = abi)

    no_patients = contract.functions.getPatientsCount().call()

    lpatient = []
    lhash = []
    message = 'No infected people'
```

```
    if request.method == 'GET':


        region = request.GET.get("city")
        postal = request.GET.get("postal")
        country = request.GET.get("country")


    counter = 0
    for i in range(0, no_patients):
        patients = contract.functions.gettPatient(i).call()
        lpatient = list(patients)
        print(lpatient)
        if lpatient[0] == 0 and lpatient[1] == postal and lpatient[4]
            == country:
            if lpatient[3] not in lhash and lpatient[2] != 'daleted':
                lhash.append(lpatient[3])
                counter = counter + 1


            message = 'Infected people in your area are'


    return render(request,'infected.html', {'countries':countries,
        'infected_people': counter, 'message': message})
```

Listing 13: Python class models.py for for hospital application

```
from __future__ import unicode_literals
from django.db import models
from django.contrib.auth.models import User
```

```
class Patient(models.Model):
    """
        Patient information in hospital database
    """
    CHOICES = [(True, 'Infected'), (False, 'Cured'), ('Suspected',
        'Suspected')]
    uid = models.AutoField(primary_key=True)
    user = models.ForeignKey(User, on_delete=models.CASCADE, default=1)
    name = models.CharField(max_length=100, blank=False, default='')
    surname = models.CharField(max_length=70, blank=False, default='')
    address = models.CharField(max_length=256, blank=False, default='')
    email = models.CharField(max_length=256, blank=False, default='')
    city = models.CharField(max_length=70, blank=False, default='')
    region = models.CharField(max_length=70, blank=False, default='')
    postal = models.CharField(max_length=70, blank=False, default='')
    country = models.CharField(max_length=70, blank=False, default='')
    phone = models.CharField(max_length=70, blank=False, default='')
    status = models.CharField(max_length=20, choices=CHOICES,
        null=True, default='')
    notes = models.TextField(null=True, blank=True)
    created_at = models.CharField(max_length=70, blank=False,
        default='')
    hashing = models.CharField(max_length=100, blank=False, default='')
```